23. The '515 patent's specification refers to an "Appendix A," which it asserts is "a generic HTML HostTemplate illustrating how Prepare and Post components are integrated into a web page," and which it asserts is "a complete working example." '515 patent at 5:17-20. But the '515 patent contains no such Appendix A.

24. In any case, the Appendix A included in the '557 and '482 patents does not disclose the structure corresponding to "code means . . . for enabling a receipt of an identification of one or more image files, video files, or audio files to associate with said account." Appendix A contains primarily HTML with a small amount of JavaScript code. It indicates that the customer (that is, the user of the web browser) will, by loading the HTML in Appendix A, cause the presumably relevant code to be downloaded from the PictureWorks server. In other words, the relevant code itself—code that would "enabl[e] a receipt of an identification of one or more image files, video files, or audio files to associate with [an] account"—may be referenced in Appendix A, but it is not disclosed there.

25. Similarly, Figures 3, 4A, and 4B do not contain executable code. Some entries, such as "fileName" and "imageName," are merely variables. Other entries are labeled as functions, but these disclose only "declarations" of function "signatures"—that is, the functions' names and the identity of the inputs (*i.e.*, parameters) that will go into them. No entry in this table discloses the "definition" of a function—that is, the operations the named function would actually perform on the inputs it is given. In other words, although the entries in these figures disclose functions' inputs and outputs, the functions themselves remain "black boxes."

26. Even if these figures disclosed executable source code—which they do not—it would make no difference here, because no piece of code purports to perform the function at issue: "enabling a receipt of an identification of one or more . . . files." The entries in the third column of Figures 3 and 4A purport to state the purpose of each piece of code. But no entry in this column relates to receiving an identification of one or more files: Of the two functions listed,

the first purportedly "[s]cales an image in place or to a temporary file," and the second purportedly "[t]ransfers [an] image and returns a status code." '515 patent, Figs. 3, 4A. In sum, the code and pseudo-code that the patent specification purports to reference fail to disclose any algorithm or other structure corresponding to the claimed "code means."

B. Other Disclosures in the Specification

27. I will now address whether any other portions of the specification disclose a structure corresponding to "code means . . . for enabling a receipt of an identification of one or more image files, video files, or audio files to associate with said account."

28. I understand from the Joint Claim Construction Statement that Summit 6 cited column 3, lines 18-54, of the '515 patent's specification as disclosing the structure corresponding to the "code means" term. I disagree that this passage discloses a structure corresponding to the "code means" term. The passage describes, among other things, something called a "media object identifier." However, the term "media object identifier" is not a term of art in software development or distributed systems. A person having ordinary skill in the art in this field would not understand "media object identifier" as corresponding to a particular structure or method of operation. Moreover, the cited passage of the specification defines the "media object identifier" solely by its function, not by any structure that would perform that function:

> The two primary components used in the Prepare and Post tools which carry out these functions are 1) the media object identifier and 2) the media sender.
>
> In general, ***the media object identifier functions to provide a graphical user interface for placing and associating a media object*** from a user's desktop onto a web page.

'515 patent at 18-20 (emphasis added). A person having ordinary skill in the art would not understand this language as describing a particular structure.

29. Later in the same passage, the specification discloses that a media object identifier may be "provid[ed] . . . as an ActiveX component" or as a "pure Java component." '515 patent at 3:35-36, 3:40-42. However, ActiveX and Java are both software ***frameworks***—languages in

which software may be written—not particular structures. In other words, disclosing that a media object identifier may be an "ActiveX component" or a "Java component" provides no more structure than disclosing that a media object identifier may be written, for example, in JavaScript—or in any other programming language.

30. The same passage also explains that a user may select an image from a list or drag and drop an image onto a media object identifier and that the media object will thereby be "associated" with the media object identifier. *See* '515 patent at 3:26-54. But describing how a user may select an image for upload does not disclose any structure "for enabling a receipt of an identification of one or more . . . files to associate with [an] account." Disclosing that a "media object identifier" accepts user input via the drag-and-drop functionality indicates only at a high level the desired behavior of such a control from a user-interface standpoint; it does not disclose structure corresponding to the claimed function.

31. I understand that Summit 6 has also cited Figures 1 and 2 as disclosing the structure corresponding to the "code means" term. Again, I disagree that the cited figures disclose any structure. Figures 1 and 2 are mock-ups of web pages, and they purport to show media object identifiers. However, these figures do not provide any description of a procedure that would communicate file names or correlate files with account information. They give no insight into what is going on "behind the scenes." Instead, they are merely illustrative of a proposed user interface. They do not indicate to a person skilled in the art what the step-by-step procedure would be for implementing the desired functionality—namely, "enabling a receipt of an identification of one or more image files, video files, or audio files to associate with [an] account." In other words, a person having ordinary skill in the art would understand from these figures that the alleged invention includes a "black box" capable of performing various functions, but he or she would not be able to glean from these figures what actually occurs inside that box.

32. I understand that Summit 6 has also cited column 4, lines 4 through 6, and lines 17 through 34, as disclosing the structure corresponding to the "code means" term. Again, I disagree that the cited passages disclose any structure. The cited passages do not relate to the recited function at all. Instead, they describe irrelevant characteristics of media object identifiers, such as the fact that "a plurality of images [may] be submitted simultaneously" and that "any number of media object identifiers may be provided on a web page." '515 patent at 4:4-6, 4:17-19.

33. I understand that Summit 6 has also cited column 5, line 13, through column 6, line 36, as disclosing the structure corresponding to the "code means" term. Again, I disagree that the cited passage discloses any structure. The cited passage describes the contents of Appendix A, which, as noted above, is not included in the specification of the '515 patent. The passage states that Appendix A discloses "a complete working example" of the Prepare and Post tools, which are described as the preferred embodiment of the alleged invention. In fact, as I explained above, Appendix A does not disclose the relevant code; it merely discloses that a potential customer could use Appendix A to download the relevant code from the PictureWorks server. *See* '515 patent at 5:29-31 ("The Initialization Section consists of a few lines of JavaScript code *that will download all of the needed Prepare and Post submission components*." (emphasis added)).

34. For the foregoing reasons, none of the portions of the specification cited by Summit 6, alone or in combination, discloses to a person of ordinary skill in the art a structure corresponding to the term "code means . . . for enabling a receipt of an identification of one or more image files, video files, or audio files for association with said account."

VI. "MEDIA OBJECT IDENTIFIER"

35. The specification of the '557 patent does not sufficiently disclose structure corresponding to the claim term "media object identifier."

36. As explained above, the term "media object identifier" is not a term of art in software development. Thus, a person having ordinary skill in the art would not understand the term as corresponding to a particular structure.

37. The specification describes the claimed "media object identifier" solely in terms of the functions it can perform, not its structure. For example, the specification states: "In general, the media object identifier ***functions to provide*** a graphical user interface for placing and associating a media object from a user's desktop onto a web page." '557 patent at 3:12-14 (emphasis added). The '557 patent claims also describe the "media object identifier" solely by what is it capable of doing—namely, receiving an "associat[ion]" of a media object and performing various types of processing on the media object, including "reducing the file size," "compressing," and "changing the file format of the media object." *See, e.g.*, '557 claims 28, 29.

38. A general-purpose computer could not perform the functions of the claimed "media object identifier" without special programming.

39. As explained in paragraphs 24 and 33, *supra*, the specification does not disclose code that creates the claimed "media object identifier"; it merely discloses that a customer may download the presumably relevant code from the PictureWorks server. *See* '557 patent at 5:39-41 ("The Initialization Section consists of a few lines of JavaScript code *that will download all of the needed Prepare and Post submission components*." (emphasis added)).

40. As explained above, the specification discloses that a media object identifier may be "provid[ed] . . . as an ActiveX component" or as a "pure Java component." '515 patent at 3:35-36, 3:40-42. However, ActiveX and Java are both software frameworks—languages in which software may be written—not particular structures. Thus, stating that a media object identifier may be an "ActiveX component" or a "Java component" discloses no more structure than stating that a media object identifier may be written, for example, in JavaScript—or in any other programming language.

41. I understand that Summit 6 has cited Figures 1-4B of the '557 patent as disclosing the structure corresponding to the term "media object identifier." For many of the reasons given above, I disagree that any of these figures, alone or in combination, sufficiently discloses the structure corresponding to the term "media object identifier."

42. Figures 1 and 2 are mock-ups of web pages, and they purport to show media object identifiers. However, these figures do not provide any description of a procedure that would perform the functions the "media object identifier" purportedly performs. In other words, they are merely illustrative of a proposed user interface; they give no insight into what is going on "behind the scenes." More particularly, these figures do not indicate to a person skilled in the art precisely what the step-by-step procedure would be for implementing the desired functionality—namely, receiving an "association" of a media object and performing various types of processing on the media object (*e.g.*, cropping, rotating, or compressing it).

43. Similarly, Figures 3, 4A, and 4B do not contain executable code. Some entries, such as "fileName" and "imageName," are merely variables. Other entries are labeled as functions, but these disclose only "declarations" of function "signatures"—that is, the functions' names and the identity of the inputs (*i.e.*, parameters) that will go into them. No entry in this table discloses the "definition" of a function—that is, the operations the named function would actually perform on the inputs it is given. In other words, although the entries in these figures disclose functions' inputs and outputs, the functions themselves remain "black boxes."

44. I understand that Summit 6 has also cited column 3, lines 9 through 46, as disclosing the structure corresponding to the claimed "media object identifier." Again, I disagree that the cited passage sufficiently discloses the corresponding structure. First, as noted above, this passage defines the media object identifier solely by its function, not by its structure:

> The two primary components used in the Prepare and Post tools which carry out these functions are 1) the media object identifier and (2) the media sender.

> In general, ***the media object identifier functions to provide a graphical user interface for placing and associating a media object*** from a user's desktop onto a web page.

'557 patent at 3:12-14 (emphasis added). A person having ordinary skill in the art would not understand this language as describing a particular structure; he or she would understand this language to mean that a "media object identifier" is any software capable of "provid[ing] a graphical user interface for placing and associating a media object from a user's desktop onto a web page."

45. Later in the same passage, the specification discloses that a media object identifier may be "provid[ed] . . . as an ActiveX component" or as a "pure Java component." '557 patent at 3:27-28, 3:35. However, ActiveX and Java are both software frameworks—languages in which software may be written—not particular structures. In other words, disclosing that a media object identifier may be an "ActiveX component" or a "Java component" provides no more structure than disclosing that a media object identifier may be written, for example, in JavaScript—or in any other programming language.

46. The same passage also explains that a user may select an image from a list or drag and drop an image onto a media object identifier and that the media object will thereby be "associated" with the media object identifier. *See* '557 patent at 3:36-46. But describing how *a user* may select an image for upload does not explain how *the software* actually receives the user's input. There are countless ways in which a person having ordinary skill in the art could program a piece of software to accept a file selection via the drag-and-drop functionality. Thus, disclosing that a "media object identifier" accepts user input via the drag-and-drop functionality indicates only at a high level the desired behavior of such a control from a user-interface standpoint; it does not indicate to a person having ordinary skill in the art *how* that function is implemented. Similarly, disclosing that a user may select an image from a list or drag and drop an image onto a media object identifier discloses nothing about how the media object identifier

performs the various types of image processing listed in the patent (*e.g.*, rotating, compressing, and changing the file format of the image).

47. I understand that Summit 6 has also cited column 3, lines 60 through 64, as disclosing the structure corresponding to the claimed "media object identifier." Again, I disagree that the cited passage sufficiently discloses the corresponding structure. The cited passage states that the claimed "media object identifier" permits users to caption their images prior to upload. Again, such a description discloses only a *function* the claimed "media object identifier" is capable of performing—namely, captioning an image—not a structure corresponding to the term "media object identifier."

48. I understand that Summit 6 has also cited column 4, lines 3 through 8 and 18 through 37, as disclosing the structure corresponding to the claimed "media object identifier." Again, I disagree that the cited passages, alone or in combination, disclose the corresponding structure. The cited passages describe the functions the "media object identifier" can perform, such as submitting "a plurality of images . . . simultaneously," and explain that "any number of media object identifiers may be provided on a web page." '557 patent at 4:4-5, 4:17-18. None of these passages gives any insight into the structure of the claimed "media object identifier."

49. I understand that Summit 6 has also cited the passage located at column 4, line 59, through column 5, line 2, as disclosing the structure corresponding to the claimed "media object identifier." Again, I disagree that the cited passage sufficiently discloses the corresponding structure. The cited passage states that "[a] key differentiator of the Prepare and Post tools is the browser, or client-side intelligence built into the tools." '557 patent at 4:59-60. Nevertheless, the remainder of the passage does not disclose the structures capable of performing the client-side processing described. Instead, it again states only the *functions* the tools can perform: "This intelligence directly provides features including those already outlined such as associating data with media objects, generating a visual representation of the media objects and generating media

object identifiers dynamically or in a pre-set manner." '557 patent at 4:60-65. Again, the structure that would permit the claimed "media object identifier" to perform these functions is not disclosed.

50. I understand that Summit 6 has also cited the passages located at column 5, lines 24 through 38, 43 through 46, and 55 through 63, and at column 6, lines 15 through 22 and 43 through 50. Again, I disagree that the cited passages, alone or in combination, disclose the structure corresponding to the claimed "media object identifier." The cited passages describe the contents of Appendix A, which, as noted above, allegedly discloses "a complete working example" of the Prepare and Post tools. In fact, as I explained above, Appendix A does not disclose the relevant code; it merely discloses that a potential customer could use Appendix A to download the code creating the claimed "media object identifier" from the PictureWorks server. *See* '557 patent at 5:39-41 ("The Initialization Section consists of a few lines of JavaScript code *that will download all of the needed Prepare and Post submission components*." (emphasis added)). These passages also describe various inputs (*i.e.*, parameters) that the claimed "media object identifier" might receive, such as the variables "Key1" and "Key2," but the claimed "media object identifier" remains a "black box."

I declare under penalty of perjury that the foregoing statements are true and correct to the best of my knowledge and belief.

Dated: December 29, 2014

By: _____
Dr. Emery Berger
Declarant

ATTACHMENT A

Emery Berger
emery@cs.umass.edu
http://www.emeryberger.com

School of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003

Research Interests

Design and implementation of programming languages, runtime systems, and operating systems, with a focus on automatically improving reliability, security, and performance.

Education

Ph.D., Computer Science, University of Texas at Austin, August 2002
Thesis: **Memory Management for High-Performance Applications**
Nominated for ACM Doctoral Dissertation Award; Advisor: Kathryn S. McKinley

M.S., Computer Science, University of Texas at Austin, December 1991
B.S., Computer Science, *cum laude*, University of Miami, May 1988

Academic Experience

Professor, University of Massachusetts Amherst, 2014–present
Associate Professor, University of Massachusetts Amherst, 2008–2014
Visiting Researcher, Microsoft Research, 2005, 2006, 2011, 2013
Associate Researcher, Barcelona Supercomputing Center, 2010–present
Visiting Professor, Universitat Politècnica de Catalunya, 2008–2009
Assistant Professor, University of Massachusetts Amherst, 2002–2008
Research Intern, Microsoft Research, Summer 2000 & 2001
Graduate Research Assistant, University of Texas at Austin, 1997–2002

Professional Experience

Systems Analyst, University of Texas at Austin, 1995–2000
Teacher, Benjamin Franklin International School, Barcelona, Spain, 1992–1994
Systems Analyst, Applied Research Laboratories: UT-Austin, 1990–1992
Instructor, The Princeton Review, Austin, Texas, 1989–1990
Teaching Assistant, University of Texas at Austin, 1989–1990
Programmer, FOCAL Informatique, Grenoble, France, Summer 1990
Programmer, Texas Instruments, Austin, Texas, 1989 – 1990
Programmer, ComPro Associates, Orlando, Florida, 1988
Programmer, Stromberg-Carlson, Inc. (now Siemens), Lake Mary, Florida, 1986
Programmer, AT&T Information Systems, Maitland, Florida, 1985
Programmer, Fetco Inc., Sanford, Florida, 1984

HONORS & AWARDS

Best Paper Award, *SurveyMan: Programming and Debugging Surveys* (OOPSLA 2014)
University of Massachusetts Exceptional Merit Award, 2014
SIGPLAN Research Highlight, *Doppio: Breaking the Browser Language Barrier*, 2014
PLDI Distinguished Artifact Award, 2014
Microsoft Software Engineering Foundation (SEIF) Award, 2013
SIGPLAN Research Highlight, *AutoMan: Integrating Human and Digital Computation*, 2013
Most Influential Paper Award, OOPSLA 2012 (10 year test of time award)
Google Research Award, 2011
ACM Senior Member, 2011
CACM Research Highlight, *Exterminator: Automatically Correcting Errors with High Probability*, December 2008
Best Paper Award, *TFS: A Transparent File System for Contributory Storage* (FAST 2007)
Lilly Teaching Fellowship, University of Massachusetts Amherst, 2006
National Science Foundation (NSF) CAREER Award, 2004 – 2007
Microsoft Research Graduate Fellowship, 2001 – 2002
Novell Corporation Fellowship, 1997 – 1998
Florida Honors Scholarship, 1984 – 1988

PUBLICATIONS: CONFERENCE PAPERS

Note: In Computer Science, unlike many other fields, conference papers are rigorously reviewed, with top conferences having low acceptance rates; publications in these conferences are considered archival and comparable to top journal papers.

Citation numbers from Google Scholar, June 2014; total citations: 2,366.

[OOPSLA 2014] **CHECKCELL: Data Debugging for Spreadsheets**, D. Barowy, D. Gochev, E. Berger. In *Proceedings of the 2014 ACM SIGPLAN Conference on Object-Oriented Programming Languages, Systems, and Applications,* October 2014. Acceptance rate: 28% (53/185). http://checkcell.org

[OOPSLA 2014] **SURVEYMAN: Programming and Automatically Debugging Surveys**, E. Tosch, E. Berger. In *Proceedings of the 2014 ACM SIGPLAN Conference on Object-Oriented Programming Languages, Systems, and Applications,* October 2014. Acceptance rate: 28% (53/185). ***Best Paper Award*** http://surveyman.org

[PLDI 2014] **DOPPIO: Breaking the Browser Language Barrier**, J. Vilk, E. Berger. In *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation,* pp. 508-518, June 2014. Acceptance rate: 18% (52/287). ***Winner of PLDI 2014 Distinguished Artifact Award; SIGPLAN Research Highlight.*** http://doppiojvm.org

[PPoPP 2014] **PREDATOR: Predictive False Sharing Detection**, T. Liu, C. Tian, Z. Hu, E. Berger. In *Proceedings of the 19th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming,* pp. 3-14, February 2014. Acceptance rate: 16% (28/179).

[ASPLOS 2013] **STABILIZER: Statistically Sound Performance Evaluation**, C. Curtsinger, E. Berger. In *Proceedings of the Eighteenth International Conference on Architectural Support for Programming Languages and Operating Systems,* pp. 219-228, March 2013. Acceptance rate: 23% (44/191). http://stabilizer-tool.org [11 citations]

[DATE 2013] **Probabilistic Timing Analysis on Conventional Cache Designs**, L. Kosmidis, C. Curtsinger, E. Quiñones, J. Abella, E. Berger, F. Cazorla. In *Proceedings of the Conference on Design, Automation and Test in Europe,* pp. 603-606, March 2013. [8 citations]

[OOPSLA 2012] **AUTOMAN: Integrating Digital and Human Computation**, D. Barowy, C. Curtsinger, E. Berger, A. McGregor. In *Proceedings of the 2012 ACM Conference on Object-Oriented Programming Languages, Systems, and Applications*, pp. 639-654, October 2012. Acceptance rate: 25% (57/228). [22 citations] *http://automan-lang.org* ***SIGPLAN Research Highlight.***

[SOSP 2011] **DTHREADS: Efficient Deterministic Multithreading**, T. Liu, C. Curtsinger, E. Berger. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles*, pp. 327-336, October 2011. Acceptance rate: 18% (28/153). [91 citations] *http://dthreads.org*

[OOSPLA 2011] **SHERIFF: Precise Detection and Automatic Mitigation of False Sharing**, T. Liu, E. Berger. In *Proceedings of the 2011 ACM Conference on Object-Oriented Programming Languages, Systems, and Applications*, pp. 3-18, October 2011. Acceptance rate: 37% (61/166). [22 citations]

[WOOT 2011] **DieHarder: Securing the Heap**, G. Novark, E. Berger. In *Proceedings of the 5th USENIX Workshop on Offensive Technologies,* August 2011 (invited paper).

[CCS 2010] **DieHarder: Securing the Heap**, G. Novark, E. Berger. In *Proceedings of the 2010 ACM Conference on Computer and Communications Security,* pp. 573-584, October 2010. Acceptance rate: 17% (55/325). [38 citations]
Inspiration for security-hardening features in Windows 8.

[OOPSLA 2009] **Grace: Safe Multithreaded Programming for C/C++**, E. Berger, T. Yang, T. Liu, G. Novark. In *Proceedings of the 2009 ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications*, pp. 81-96, October 2009. Acceptance rate: 17% (25/144). [190 citations]

[ECRTS 2009] **Using Randomized Caches in Real-Time Systems**,
E. Quiñones, E. Berger, G. Bernat, F. Cazorla. In *Proceedings of the 21st IEEE Euromicro Conference on Real-Time Systems*, pp. 129-138, June 2009. Acceptance rate: 25% (26/102). [23 citations]

[PLDI 2009] **Efficiently and Precisely Locating Memory Leaks and Bloat**, G. Novark, E. Berger, B. Zorn. In *Proceedings of the 2009 ACM Conference on Programming Language Design and Implementation*, pp. 397-407, June 2009. Acceptance rate: 21% (41/194). [37 citations]

[OSDI 2008] **Redline: First Class Support for Interactivity in Commodity Operating Systems**, T. Yang, T. Liu, E. Berger, S. Kaplan, J. Moss. In *Proceedings of the 8th USENIX Symposium on Operating System Design and Implementation*, pp. 73-86, December 2008. Acceptance rate: 13% (26/193). [53 citations]

[ASPLOS 2008] **Archipelago: Trading Address Space for Reliability and Security**, V. Lvin, G. Novark, E. Berger, and B. Zorn. In Proceedings of the Thirteenth International Conference on Architectural Support for Programming Languages and Operating Systems-XIII, 10 pages, March 2008. Acceptance rate: 24% (31/127). [40 citations]

[SenSys 2007] **Eon: A Language and Runtime System for Perpetual Systems**, J. Sorber, A. Kostadinov, M. Brennan, M. Garber, M. Corner, and E. Berger. In *Proceedings of the 5th ACM Conference on Embedded Networked Sensor Systems*, pp. 161-174, November 2007. Acceptance rate: 16%. [120 citations]

[PLDI 2007] **Exterminator: Automatically Correcting Memory Errors with High Probability**, G. Novark, E. Berger, and B. Zorn. In *Proceedings of the 2007 ACM Conference on Programming Language Design and Implementation*, pp. 1-11, June 2007. Acceptance rate: 25% (45/178).
Selected as a CACM Research Highlight.
[114 citations]

[FAST 2007] **TFS: A Transparent File System for Contributory Storage**, J. Cipar, M. Corner, E. Berger. In *Proceedings of the Fifth USENIX Conference on File and Storage Technologies*, pp. 215-229, February 2007. Acceptance rate: 20%. ***Best paper award.*** [26 citations]

[OSDI 2006] **CRAMM: Virtual Memory Support for Garbage-Collected Applications**, T. Yang, E. Berger, S. Kaplan, J. E. B. Moss. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation*, pp. 103-116, November 2006. Acceptance rate: 18% (27/150). [67 citations]

[PLDI 2006] **DieHard: Probabilistic Memory Safety for Unsafe Languages**, E. Berger and B. Zorn. In *Proceedings of the 2006 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 158-167, June 2006. Acceptance rate: 21% (36/174).
Directly inspired the design of the Windows Fault-Tolerant Heap.
[247 citations]

[USENIX 2006] **Flux: A Language for Programming High-Performance Servers**, B. Burns, K. Grimaldi, A. Kostadinov, E. Berger and M. Corner. In *Proceedings of the USENIX 2006 Annual Technical Conference*, pp. 129-142, May 2006. Full paper acceptance rate: 13.7% (21/153). [38 citations]

[USENIX 2006] **Transparent Contribution of Memory**, J. Cipar, M. Corner, E. Berger. In *Proceedings of the USENIX 2006 Annual Technical Conference*, pp. 109-114, May 2006. Acceptance rate: 18.5%. [13 citations]

[OOPSLA 2005] **Quantifying the Performance of Garbage Collection vs. Explicit Memory Management**, M. Hertz and E. Berger. In *Proceedings of the 2005 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications*, pp. 313-326, October 2005. Acceptance rate: 18% (32/174). [78 citations]

[PLDI 2005] **Garbage Collection Without Paging**, M. Hertz, Y. Feng, and E. Berger. In *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 143-153, June 2005. Acceptance rate: 21% (28/135). [69 citations]

[MSP 2005] **A Locality-Improving Dynamic Memory Allocator**, Y. Feng, E. Berger. In *3rd Annual ACM SIGPLAN Workshop on Memory Systems Performance*, pp. 68-77, June 2005. Acceptance rate: 33%. [43 citations]

[OOPSLA 2004] **MC^2: High-Performance Garbage Collection for Memory Constrained Environments**, N. Sachindran, J.E.B. Moss and E. Berger. In *Proceedings of the 2004 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications*, pp. 81-98, October 2004. Acceptance rate: 15%. [38 citations]

[ISMM 2004] **Automatic Heap Sizing: Taking Real Memory into Account**, T. Yang, M. Hertz, E. Berger, S. Kaplan, J.E.B. Moss. In *Proceedings of the 2004 ACM SIGPLAN International Symposium on Memory Management*, pp. 61-72, October 2004. Acceptance rate: 34% (15/43). [49 citations]

[OOPSLA 2002] **Reconsidering Custom Memory Allocation**, E. Berger, B. Zorn and K. S. McKinley. In *Proceedings of the 2002 ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications*, pp. 1-12, November 2002. Acceptance rate: 20% (25/125), impact: top 2.29%. ***Winner of 2012 OOPSLA Most Influential Paper Award.*** [158 citations]

[PLDI 2001] **Composing High-Performance Memory Allocators**, E. Berger, B. Zorn and K. S. McKinley. In *Proceedings of the 2001 ACM SIGPLAN Conference on Programming Language Design and Implementation*, pp. 114-124, June 2001. Acceptance rate: 20% (30/144), impact: top 0.24%. [120 citations] *http://www.heaplayers.org*

[PPSC 2001] **Customizing Software Libraries for Performance Portability**, S. Guyer, E. Berger, and C. Lin. In *10th SIAM Conference on Parallel Processing for Scientific Computing,* March 2001 [5 citations]

[ASPLOS-IX] **Hoard: A Scalable Allocator for Multithreaded Applications**, E. Berger, K. S. McKinley, R. Blumofe and P. Wilson. In *The Ninth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 117-128, November 2000. Acceptance rate: 21% (24/114). *http://www.hoard.org* [352 citations]
Algorithm adopted by Mac OS X and IBM; numerous commercial users. Variant of Hoard awarded test-of-time award at PLDI 2014.

PUBLICATIONS: JOURNAL ARTICLES

[TECS 2013] **PROARTIS: Probabilistically Analysable Real-Time Systems**, F. Cazorla, E. Quiñones, T. Vardanega, L. Cucu, B. Triquet, G. Bernat, E. Berger, J. Abella, F. Wartel, M. Houston, L. Santinelli, L. Kosmidis, C. Lo, D. Maxim. In *ACM Transactions on Embedded Computing Systems*, Volume 12, Issue 2s (94 pages), May 2013. [35 citations]

[CACM 2012] **Software Needs Seatbelts and Airbags**, E. Berger. In *Communications of the ACM*, Volume 55, Issue 9, pp. 48-53, September 2012.

[CACM 2008] **Exterminator: Automatically Correcting Memory Errors with High Probability**, G. Novark, E. Berger, B. Zorn. In *Communications of the ACM (Research Highlight),* pp. 87-95, December 2008.

[TOS 2007] **TFS: A Transparent File System for Contributory Storage**, J. Cipar, M. Corner, E. Berger. In *ACM Transactions on Storage*, Volume 3, Issue 3, Article 12 (26 pages), October 2007. [28 citations]

[IJHPCA 2000] **Compositional Development of Performance Models in POEMS**, J.C. Browne, E. Berger, and A. Dube. In *International Journal of High Performance Computing Applications*, Sage Science Press, Volume 14, Number 4 (pp. 283-291), Winter 2000 [22 citations]

[IJNME 1998] **A Fast Solution Method for Three-Dimensional Many-Particle Problems of Linear Elasticity**, Y. Fu, K. Klimkowski, G. Rodin, E. Berger, J. C. Browne, J. Singer, R. van de Geijn, and K. Vemaganti. In *International Journal for Numerical Methods in Engineering*, Volume 42, 1998 [135 citations]

PUBLICATIONS: TECHNICAL REPORTS (NOT PUBLISHED ELSEWHERE)

Efficient Probabilistic Memory Safety, E. Berger and B. Zorn. UMass CS Technical Report TR-07-17, March 2007.

HeapShield: Library-Based Heap Overflow Protection for Free, E. Berger. UMass CS Technical Report TR-06-28, June 2006. [6 citations]

Custom Object Layout for Garbage-Collected Languages, G. Novark, T. Strohman, and E. Berger. UMass CS Technical Report, TR-06-06, January 2006. [4 citations]

Optimizing Shell Scripting Languages, E. Berger. UMass CS Technical Report TR-03-09, November 2003.

Detecting Errors with Whole-Program Configurable Dataflow Analysis, S. Guyer, E. Berger, and C. Lin. UTCS Technical Report TR-02-04, January 2002. [16 citations]

FP + OOP = Haskell, E. Berger. UTCS Technical Report TR-92-30, January 1992. [7 citations]

PATENTS

US Patent #7802232, E. Berger and B. Zorn, "Software Robustness Through Search for Robust Runtime Implementations", 9/21/2010 [4 citations]

E. Berger and B. Zorn, "Software Variation for Robustness Through Randomized Execution Contexts", application filed 3/31/2006

RESEARCH SUPPORT

E. Berger (PI), S. Freund (PI), **XPS: SDA: SCORE: Scalability-Oriented Optimization**, National Science Foundation, $648,000, 9/2015-8/2019

E. Berger (PI), Alexandra Meliou, **EAGER: Data Debugging**, National Science Foundation, CCF- 1349784, $150,000, 9/2013-3/2015

E. Berger (PI), **EAGER: Programming the Crowd**, National Science Foundation, CCF-1144520, $300,002, 8/2011-8/2013

E. Berger (PI), **CheckCell: Data Debugging for Spreadsheets**, Microsoft Software Engineering Innovation Foundation (SEIF) Award, $25,000, 3/2013-unlimited.

E. Berger (PI), **Causal Profiling**, Google Research Award, $50,000, 12/2011-unlimited.

E. Berger (PI), Amazon AWS Teaching Grant, $2,000, 12/2011-unlimited.

E. Berger (PI), **Reliable Performance**, National Science Foundation, CCF-1012195 (collaborative with D. Jiménez, UT-San Antonio), $550,000, 8/1/2010-7/31/2012.

E. Berger (PI), **Perpetually-Available Software Systems**, Gigascale Systems Research Center, $315,000, 11/1/2009-10/31/2012

E. Berger (PI), **PASS: Perpetually-Available Software Systems**, National Science Foundation CCF-0910883 (collaborative with K. McKinley, UT-Austin and M. Hicks, Maryland),

$639,420, 8/1/2009-7/31/2013

F. Cazorla (PI) (Barcelona Supercomputing Center), Co-PIs: Emery Berger, Guillem Bernat (Rapitime Systems), Tullio Vardanega (University of Padua), Liliana Cucu (INRIA), Benoit Triquet (Airbus). **PROARTIS – PRObabilistic Analyzable Real-Time Systems**. €1,810,621 (2/1/2010 – 1/31/2013), European Commission FP7-ICT-2009-4, Proposal 249100.
E. Berger (PI)**, Using Multiple Cores to Improve Reliability and Security**, Intel Research Grant, $30,000, April 2007 – unlimited

E. Berger (PI), **Probabilistically Correct Execution: Hardening Applications Against Error and Attack,** National Science Foundation CNS-0615211, $300,000, 9/15/06 – 9/14/09

R. Manmatha (PI), J. Allan, E. Berger, D. Kulp, **Cluster Acquisition for Computational Research into Large Scale Data Rich Problems**, National Science Foundation CNS-0619337, $350,000, 9/1/06 – 8/31/08

E. Berger (PI)**, Using Multiple Cores to Improve Reliability and Security**, Intel Research Grant, $30,000, 4/06 – unlimited

E. Berger**,** Microsoft Research Gift, $30,000, September 2005

B. Levine (PI), E. Berger, M. Corner. **Building IA Capacity at UMass Amherst**, **DoD**, $130,000, 9/1/05 – 12/31/06

E. Berger (PI), **Cooperative System Support for Robust High Performance**,
National Science Foundation CAREER Award CNS-0347339,
$477,000, 6/1/04 – 5/31/09

SELECTED SOFTWARE

All software systems available at plasma.cs.umass.edu/emery/software.

The Hoard scalable memory allocator. Hoard is a widely-deployed memory management library that provably improves the scalability and performance of multithreaded applications. Commercial users include AOL, British Telecom, Business Objects (SAP), Cisco, Credit Suisse First Boston, Entrust, Kamakura Corporation, Novell, Open Text, Pervasive Software, Philips, Plath GmbH, Reuters, Royal Bank of Canada, Quest, Sonus Networks, TIBCO, and VSNL International. The Mac OS X and IBM memory allocators are directly based on Hoard's design. (www.hoard.org, over 100,000 downloads)

DieHard. A system that transparently improves the reliability and security of C/C++ applications. (www.diehard-software.org, over 20,000 downloads). DieHard was the direct inspiration for the Fault-Tolerant Heap incorporated in Windows 7; DieHarder, a secure variant, inspired the security-hardening features in Windows 8.

Heap Layers. A flexible infrastructure for composing high-performance general and custom memory managers. (www.heaplayers.org). Hoard, DieHard, and DieHarder were built using Heap Layers.

TEACHING EXPERIENCE

UNIVERSITY OF MASSACHUSETTS, *Department of Computer Science*

CMPSCI 630:	Graduate Systems, 2011-2014 (new core class)
CMPSCI 230:	Computer Systems Principles, 2010-2011 (new required class)
CMPSCI 691W:	Parallel & Concurrent Programming, Spring 2006 (new)
CMPSCI 691S:	Hot Topics in Programming Languages & Systems, Fall 2005
CMPSCI 691R:	Topics in Runtime Systems, Fall 2004 (new)
CMPSCI 377:	Operating Systems (fully revised), Fall 2003, Fall 2004, Fall 2005, Spring 2006, Fall 2007, Fall 2009
CMPSCI 710:	Advanced Compiler Techniques (fully revised) Spring 2003, Spring 2004
CMPSCI 691P:	Robust Software Systems, Fall 2002 (new)

HIPEAC SUMMER SCHOOL: Ninth International Summer School on Advanced Computer Architecture and Compilation for High-Performance and Embedded Systems
(course title: "Software Fault Tolerance and Correction")
Fiuggi, Italy, July 2013

Garbage Collection & Memory Management Summer School
UNIVERSITY OF KENT AT CANTERBURY, UK, July 2004

PROFESSIONAL ACTIVITIES & SERVICE

Associate Editor, ACM Transactions on Programming Languages and Systems (TOPLAS), 2007–present

Program Chair, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI), 2016

Organizer and Co-Program Chair, First Workshop on Approximate and Probabilistic Computing (APPROX), 2014

Co-Program Chair, USENIX Workshop on Hot Topics in Parallelism (HotPar), 2013

Organizer and Program Chair, Workshop on Determinism and Correctness in Parallelism (WoDet 3) 2012

Co-Program Chair and Program Committee Member, ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE), 2010

General Chair and Program Committee Member, ACM SIGPLAN Workshop on Memory Systems Performance & Correctness (MSPC) 2008

Publicity Chair, ACM Conference on Object-Oriented Programming Systems, Languages, and Applications (SPLASH/OOPSLA), 2013

Program Committee Member, USENIX Security 2014

Program Committee Member, International Symposium on Memory Management (ISMM) 2014

Program Committee Member, Workshop on Determinism and Correctness in Parallelism (WoDet) 2014

Program Committee Member, International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS) 2014

External Review Committee Member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI) 2015

External Review Committee Member, International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS) 2015

External Review Committee Member, ACM SIGPLAN Conference on Principles of Programming Languages (POPL) 2014

External Review Committee Member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI) 2014

External Review Committee Member, 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI) 2014

External Review Committee Member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI) 2014

Program Committee Member, ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI) 2013

External Review Committee Member, ACM SIGPLAN Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA) 2013

External Review Committee Member, ACM SIGPLAN Conference on Object-Oriented Programming Languages, Systems, and Applications (OOPSLA) 2012

Program Committee Member, Fifth Annual International Systems and Storage Conference (Systor 2012)

Program Committee Member, USENIX Conference on Hot Topics in Parallelism (HotPar) 2012

Program Committee Member, ACM Symposium on Principles and Practice of Parallel Programming (PPoPP) 2012

External Review Committee Member, ACM SIGPLAN Conference on Principles of Programming Languages (POPL) 2012

Program Committee Member, Workshop on Deterministic Parallelism (WoDet) 2011

Program Committee Member, ACM Conference on Computer and Communications Security (CCS) 2010

Program Committee Member, 9th USENIX Symposium on Operating Systems Design and Implementation (OSDI) 2010

Program Committee Member, Fifteenth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS) 2010

External Review Committee Member, ACM SIGPLAN Conference on Programming Languages Design and Implementation (PLDI) 2010

Program Committee Member, ACM SIGPLAN Conference on Programming Languages Design and Implementation (PLDI) 2008

Program Committee Member, ACM Symposium on Principles and Practice of Parallel Programming (PPoPP) 2008

Program Committee Member, ACM SIGPLAN International Symposium on Memory Management (ISMM) 2007

Program Committee Member, Workshop on Linguistic Support for Modern Operating Systems (PLOS) 2007

Program Committee Member, ACM SIGPLAN Conference on Programming Languages Design and Implementation (PLDI) 2007, Student Research Competition

Program Committee Member, 16th International Conference on Compiler Construction (CC) 2007

Program Committee Member, ACM SIGPLAN Workshop on Memory Systems Performance & Correctness (MSPC) 2006

Program Committee Member, ACM SIGPLAN Conference on Programming Languages Design and Implementation (PLDI) 2004

Program Committee Member, ACM SIGPLAN International Symposium on Memory Management (ISMM) 2004

Program Committee Member, Fourth International Workshop on Software and Performance (WOSP) 2004

Editorial Board Member, Science of Programming, Special Issue on Memory Management

Reviewer: ICSE, ASPLOS, HPCA, ICFP, ICPP, INTERACT, IPDPS, ISMM, ISPASS, JISE, JPDC,

OOPSLA, PACT, PLDI, POPL, SPAA, SP&E, TOPLAS, IEEE TPDS, IEEE TOC

Panelist: National Science Foundation, 2006, 2007, 2012

President, **GRACS**, the computer science graduate student association of the University of Texas at Austin, 1995 – 1997

Developed **TEXbooks**, the official textbook site for the University of Texas at Austin

PH.D. STUDENTS SUPERVISED

Dan Barowy ***(Finalist for Microsoft Research Fellowship)***
Charlie Curtsinger ***(Winner of Google Research Fellowship)***
Emma Tosch ***(Winner PLDI 2014 Student Research Competition; Best Paper OOPSLA 2014)***
Amee Trivedi
John Vilk ***(Winner of PLDI 2014 Distinguished Artifact Award)***

Matthew Hertz (Associate Professor, Canisius College)
Tongping Liu (Assistant Professor, University of Texas at San Antonio)
Gene Novark (Morgan Stanley)
Ting Yang (***Winner of UMass CS Outstanding Dissertation Award.*** First job: Intel Corp., now at Facebook)

MASTER'S STUDENTS SUPERVISED

Dimitar Gochev (2012-14)
Nitin Gupta (2010-12) (now at Intel)
Justin Aquadro (2009-11) (now at Atalasoft)
Divya Krishnan, M.S. thesis advisor (2007-9) (now at Cisco)
Jim Cipar (co-advised with Mark Corner) (2005-7) (now at Carnegie-Mellon University)
Vitaliy Lvin, M.S. thesis advisor (2006-7) (now at Google)
Yong Yuan, M.S. thesis advisor (2003-4)
Yi (Eric) Feng, M.S. thesis advisor (2002-4), (now at Google)
Pritesh Sharma, M.S. thesis advisor (2002-3)

UNDERGRADUATE STUDENTS SUPERVISED

Molly McMahon
Justin Aquadro
Duane Bailey
Jacob Evans
John Gaquin
Ali Shah
Gabriel Tarasuk-Levin (Hampshire College)
Matthew Meehan
Kevin Grimaldi
Alex Kostadinov
Laura Strickman (Amherst College)

Ana Mocanu (Amherst College), senior thesis co-advisor (2002-3)

OTHER STUDENT SUPERVISION

Santosh Nagarakatte (University of Pennsylvania), external member Ph.D. committee
Amittai Aviram (Yale University), external member Ph.D. committee
Benjamin Ransford, member Ph.D. committee
Trevor Strohman, member Ph.D. committee (Google)
Ed Walters, member Ph.D. committee
Brendan Burns, independent study supervisor (Google)
Bhuvan Urgaonkar, member Ph.D. committee (Penn. State)
John Cavazos, member Ph.D. committee (Univ. of Delaware)
Abhishek Chandra, member Ph.D. committee (U. Minnesota)
Asjad Khan, member Ph.D. committee
Naren Sachindran, member Ph.D. committee (IBM India)
Ying Gong, Synthesis project co-advisor (2003-4)
Andrew Kielbasinski, member Honors Culminating Experience committee (2003-4)
Virginie Guionnet (Universite de La Rochelle), co-advisor (2002-3)

DEPARTMENTAL & UNIVERSITY SERVICE

Chair: Admissions Committee, 2013-15.
Member: University Academic Honesty Board, 2013-14.
Member: Faculty Hiring Committee, 2013-14.
Chair: Public Relations Committee, 2012-15; ex-officio member Development Committee, Strategic Planning Committee 2012-15.
Chair: Faculty Hiring Committee, 2011-12.
Chair: Distinguished Lecture Series Committee, 2010-11.
Chair: Admissions Committee, 2009-10.
Co-chair: Admissions Committee, 2003-5.
Organizer: UMass CS Systems Lunch: http://plasma.cs.umass.edu/emery/systems-lunch
Representative: University Library Committee, 2006-7.
Member: Awards Committee (2010-11), Strategic Planning Committee (2007), Website Committee (2006-7), Curriculum Committee (2005-6), Faculty Recruiting Committee (2004-7), Personnel Committee (2003-4), Admissions Committee (2002-2003), Computing Committee (2002-3), Ad Hoc Graduate Curriculum committee (2002-3), Outreach Committee (2006)
Panel member: Professionalism Seminar on Job Hunting
Panel member: Professionalism Seminar on Ethics
Speaker: Lab Description Seminar (2002, 2004, 2005, 2006)
Moderator: Panel Discussion, CS Saturday (2005)

INVITED TALKS

"Programming Language Technology for the Sciences"
UC San Diego, December 2014 ***(Distinguished Colloquium Speaker)***
UC Berkeley, December 2014

"CheckCell: Data Debugging for Spreadsheets"
Microsoft Research Faculty Summit, August 2014

"Doppio: Breaking the Browser Language Barrier"
Samsung Research, April 2014
Google, July 2014

"Stabilizer: Statistically Sound Performance Evaluation"
Yahoo! Labs, July 2014
Facebook, July 2014
Microsoft Research, August 2013
Google, April 2013

"Programming with People: Integrating Human and Digital Computation"
Tufts University, October 2013
Brown University, October 2013
Microsoft Research, August 2013
Telefónica Research, July 2013
University of Texas at Austin, May 2013
Stanford University, April 2013
ETAPS Conference, March 2013 ***(Conference Keynote Speaker)***
ETH-Zürich, December 2012 ***(Distinguished Colloquium Speaker)***
Goethe University (Frankfurt), November 2012
University of Marburg, November 2012
University of California-Berkeley, October 2012
University of Pennsylvania, October 2012
Yahoo! Research (Barcelona), July 2012
Universitat Politecnica de Catalunya (UPC), July 2012
University of Southern Maine (NEPLS), May 2012
Williams College, May 2012
University of Ghent, March 2012

"AutoMan: A Platform for Integrating Human and Digital Computation"
POPL "Off-the-Beaten Track" Workshop, January 2012

"Multithreaded Programming for Mere Mortals"
Université de Pierre et Marie Curie (Jussieu), October 2012
Universitat Politecnica de Catalunya (UPC), July 2011
Intel Corporation, June 2011
University of Texas at Austin, January 2011

"DieHarder: Securing the Heap"
http://static.usenix.org/multimedia/woot11novark
ASPLOS 2011 PC Symposium, Seattle, WA, October 2010
Talk presentation at CCS 2010, Chicago, IL, October 2010

"Sheriff: Detecting and Eliminating False Sharing"
http://research.microsoft.com/apps/video/default.aspx?id=137990

Microsoft Research (Redmond), September 2010
Gigascale Systems Research Center (GSRC) Annual Review,
San José, CA, September 2010

"Grace: Safe Multithreaded Programming in C/C++"
http://research.microsoft.com/apps/video/default.aspx?id=115873

University of Wisconsin-Madison, October 2010
University of California-Berkeley, September 2010
University of Washington, December 2009
Microsoft Research (Redmond), December 2009
Microsoft Research (Cambridge, UK), August 2009
Universidad Complutense de Madrid (Madrid, Spain), May 2009
Universitat Politècnica de Catalunya (Barcelona, Spain), September 2008

"Cooperative Memory Management: Avoiding Paging and Improving Performance"
Sun Microsystems (Boston), December 2007

"Automatically Tolerating and Correcting Memory Errors":
Universitat Politècnica de Catalunya (Barcelona, Spain), April 2008
University of Wisconsin-Madison, November 2007
Google (Seattle), August 2007
Rutgers University, May 2007
Williams College, April 2007

"Exploiting Multiple Cores Now: Scalability and Reliability for Off-the-shelf Software":
Universitat Politècnica de Catalunya (Barcelona, Spain), December 2006
Purdue University, November 2006
University of Illinois (Urbana-Champaign), November 2006
University of California Berkeley, October 2006
University of California Los Angeles, October 2006
Harvard University, September 2006
Brown University, September 2006
Sun Microsystems, September 2006
University of Washington (Seattle), August 2006
Microsoft Research, August 2006
Intel Research (Santa Clara), August 2006
Intel Corporation (Portland), August 2006

"DieHard: Probabilistic Memory Safety for Unsafe Languages":
École Polytechnique Fédérale de Lausanne (Switzerland), December 2006
PLDI 2006 (Ottawa, Canada), June 2006
University of Maryland – College Park, May 2006
University of Texas at Austin, May 2006
Massachusetts Institute of Technology, March 2006
Brown University, October 2005

"Quantifying the Performance of Garbage Collection vs. Explicit Memory Management",

University of Washington, July 2005

"Garbage Collection without Paging", *Microsoft Research*, July 2005

"Memory Management for High-Performance Applications":
Brown University, October 2004
Georgia Institute of Technology, November 2003
Rutgers University, March 2003

"Cooperative Memory Management,"*Microsoft Research*, January 2004

"But I Didn't Want My Java Decaf!" (with Eliot Moss), *New England Programming Languages Symposium*, University of Massachusetts, Amherst, MA, June 2003

"Hoard: A Scalable Memory Allocator for Multithreaded Programs," *Ninth Workshop on Scalable Shared Memory Multiprocessors*, Vancouver, BC, Canada, June 2000

"The Hoard Multiprocessor Memory Allocator," *Microsoft Research*, March 2000.

MISCELLANEOUS

Natural languages: native English speaker, speaker of Spanish, Catalan, and French.
US and EU citizen (Hungary)

US006035323A

United States Patent [19]

Narayen et al.

[11] **Patent Number: 6,035,323**

[45] **Date of Patent: Mar. 7, 2000**

[54] **METHODS AND APPARATUSES FOR DISTRIBUTING A COLLECTION OF DIGITAL MEDIA OVER A NETWORK WITH AUTOMATIC GENERATION OF PRESENTABLE MEDIA**

[75] Inventors: **Shantanu Narayen**, Sunnyvale; **Wu Wang**, Los Altos; **Steve Morris**, Palo Alto; **Chan Chiu**, Sunnyvale; **Cecilia Zhao**, Newark; **Aditya Khosla**, Mountain View; **James Lei**, Cupertino; **Prasad Kongara**, San Jose, all of Calif.

[73] Assignee: **Pictra, Inc.**, Sunnyvale, Calif.

[21] Appl. No.: **08/957,224**

[22] Filed: **Oct. 24, 1997**

[51] **Int. Cl.**[7] **G06F 13/00**

[52] **U.S. Cl.** **709/201**; 709/232; 709/246; 707/501

[58] **Field of Search** 709/200, 201, 709/231, 232, 246; 707/501

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,608,874	3/1997	Ogawa et al. .	
5,706,502	1/1998	Foley et al. .	
5,710,883	1/1998	Honz et al. .	
5,715,397	2/1998	Ogawa et al. .	
5,745,360	4/1998	Leone et al. .	
5,752,022	5/1998	Chiu et al. .	
5,778,367	7/1998	Wesinger et al.	707/10
5,802,299	9/1998	Logan et al.	709/218
5,841,432	11/1998	Carmel et al.	345/302
5,845,084	12/1998	Cordell et al.	709/234
5,862,346	1/1999	Kley et al.	709/243
5,870,552	2/1999	Dozier et al.	709/219
5,890,170	3/1999	Sidana	707/501
5,892,909	4/1999	Grasso et al.	709/201

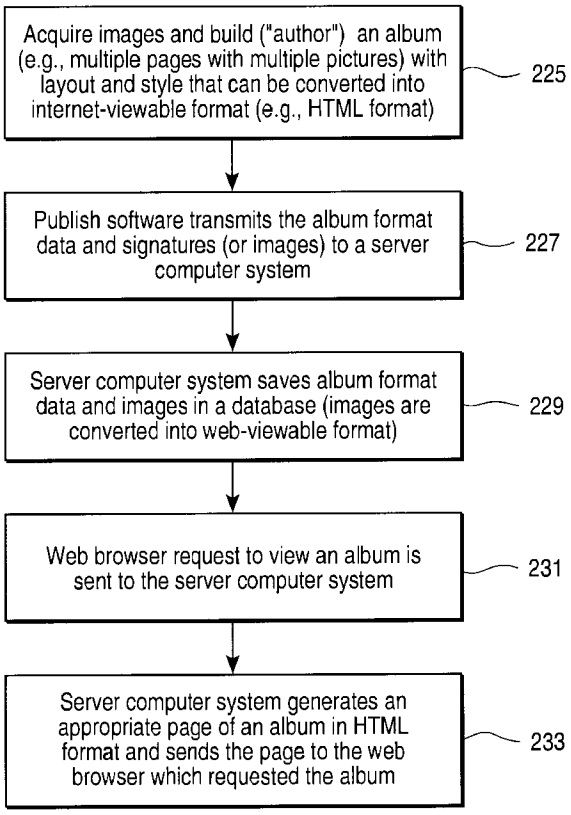
Primary Examiner—Krisna Lim

Attorney, Agent, or Firm—Blakely, Sokoloff, Taylor & Zafman

[57] **ABSTRACT**

Methods and apparatuses for publishing a collection of digital media on a network. In one example of a method, a client digital processing system generates a collection of digital media and transmits collection information, which describes the collection of digital media, to a server digital processing system. From the collection information, a plurality of presentable media is automatically generated; each of these presentable media is capable of being presented to other client digital processing systems which are coupled to the network. In this one example, the network is operating according to a hypertext transfer protocol. In this one example, the client and server systems are programmed to interact together such that the presentable media is automatically generated.

38 Claims, 21 Drawing Sheets



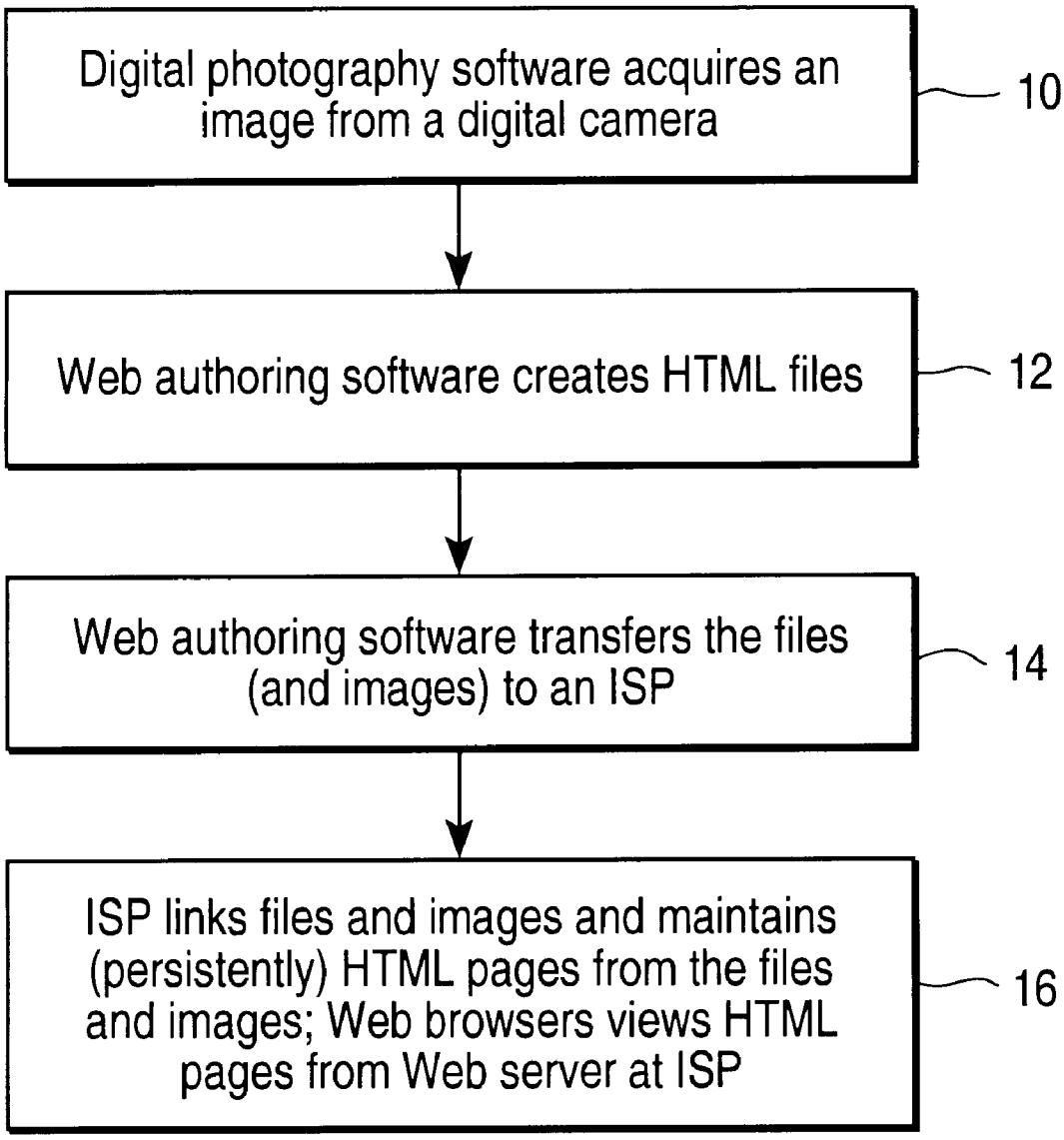


FIG. 1
(PRIOR ART)

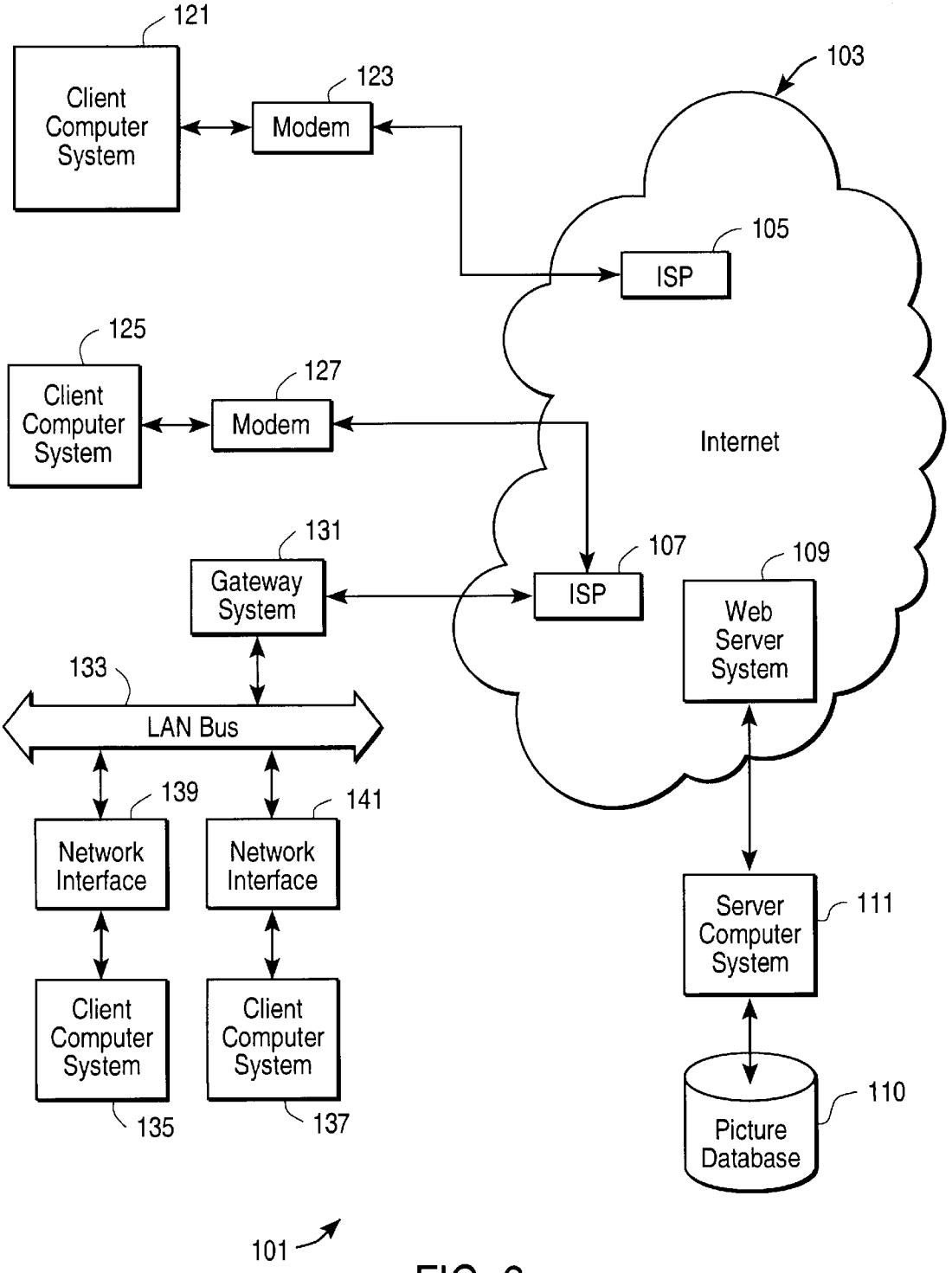

121
Client Computer System
123
Modem
103
105
ISP
125
Client Computer System
127
Modem
Internet
131
Gateway System
107
ISP
109
Web Server System
133
LAN Bus
139
141
Network Interface
Network Interface
Server Computer System
111
Client Computer System
Client Computer System
110
Picture Database
135
137
101

FIG. 2

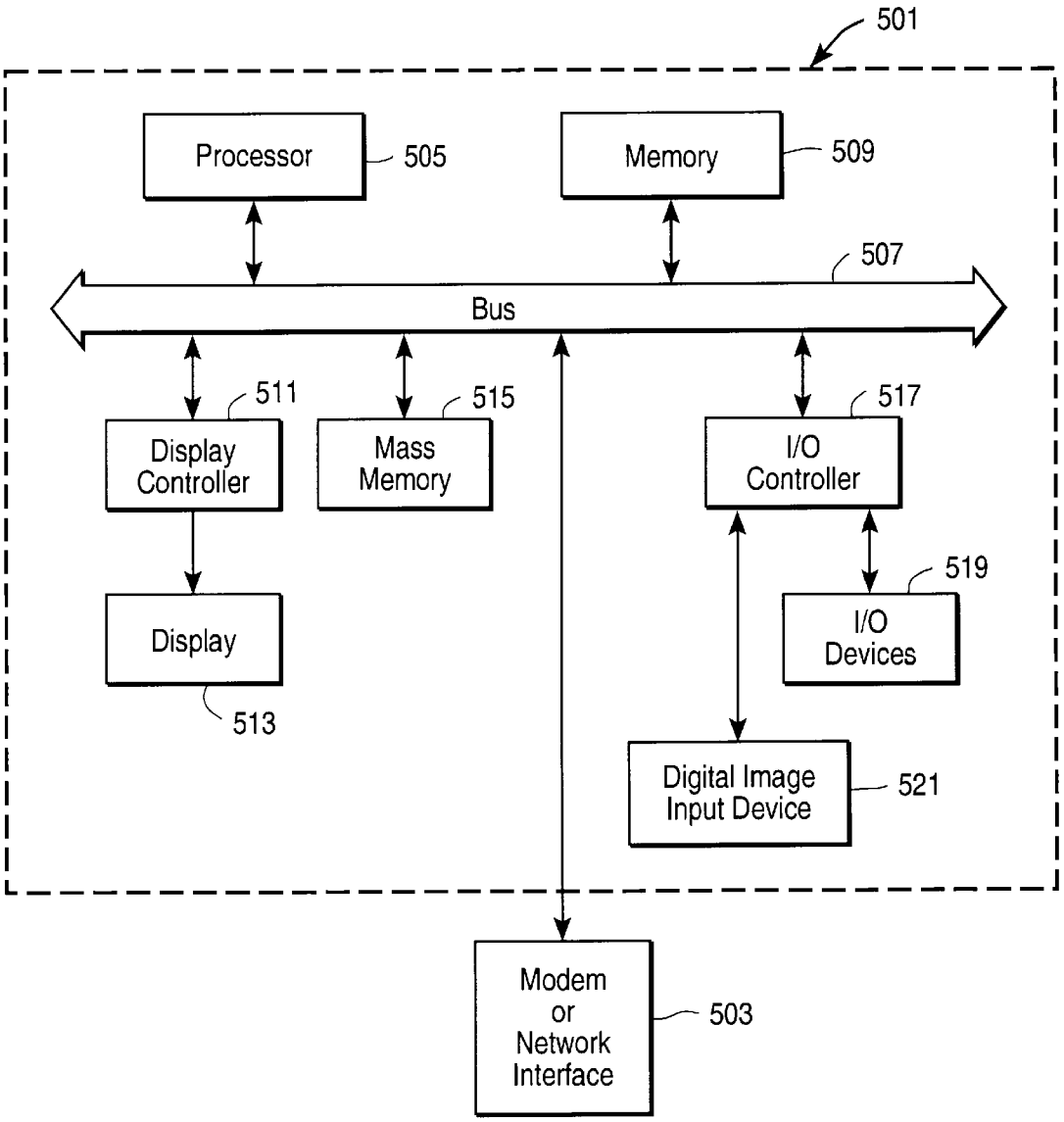

501
Processor
505
Memory
509
507
Bus
511
515
517
Display Controller
Mass Memory
I/O Controller
519
Display
I/O Devices
513
Digital Image Input Device
521
Modem or Network Interface
503

FIG. 3

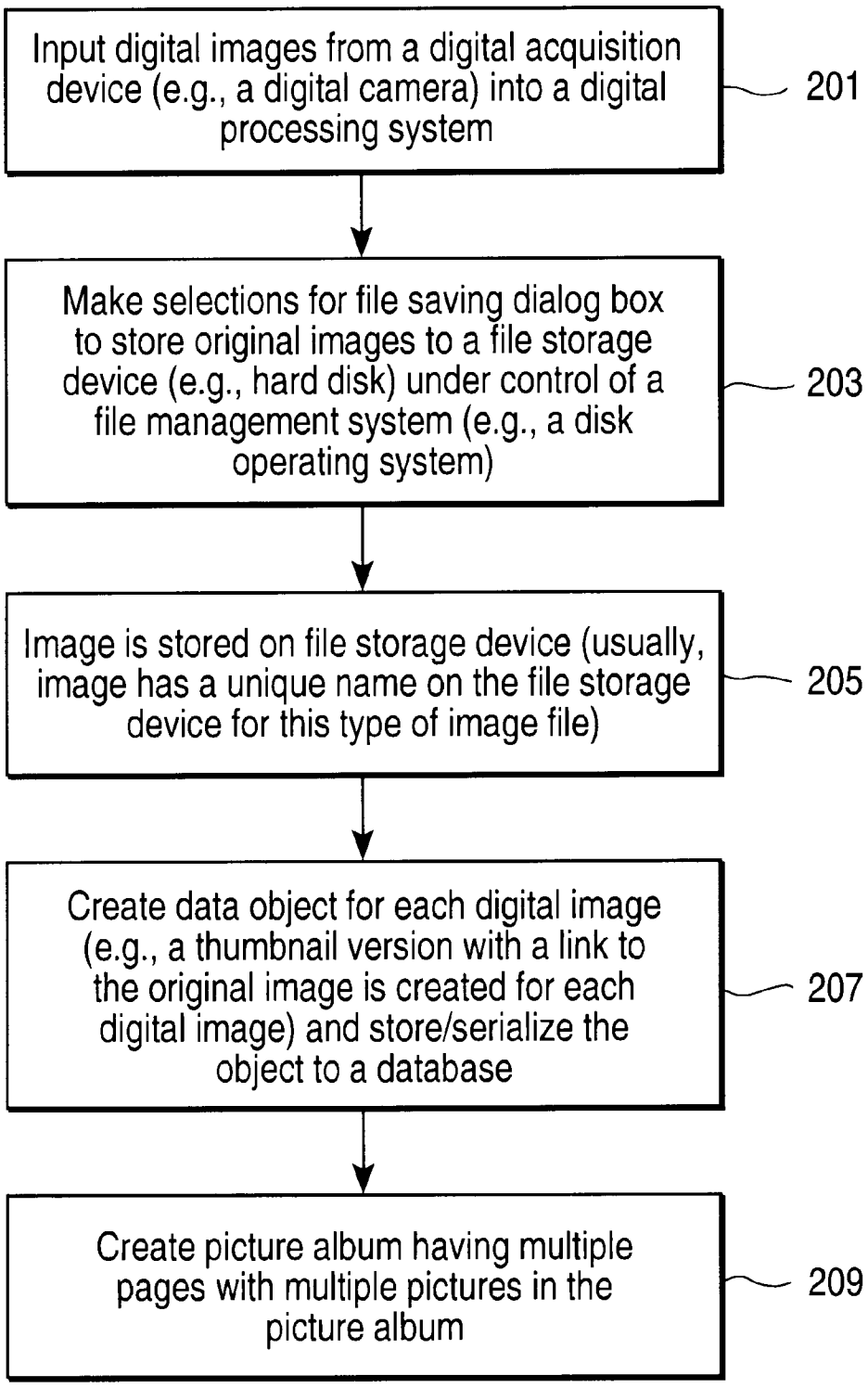

Input digital images from a digital acquisition device (e.g., a digital camera) into a digital processing system
201
Make selections for file saving dialog box to store original images to a file storage device (e.g., hard disk) under control of a file management system (e.g., a disk operating system)
203
Image is stored on file storage device (usually, image has a unique name on the file storage device for this type of image file)
205
Create data object for each digital image (e.g., a thumbnail version with a link to the original image is created for each digital image) and store/serialize the object to a database
207
Create picture album having multiple pages with multiple pictures in the picture album
209

FIG. 4

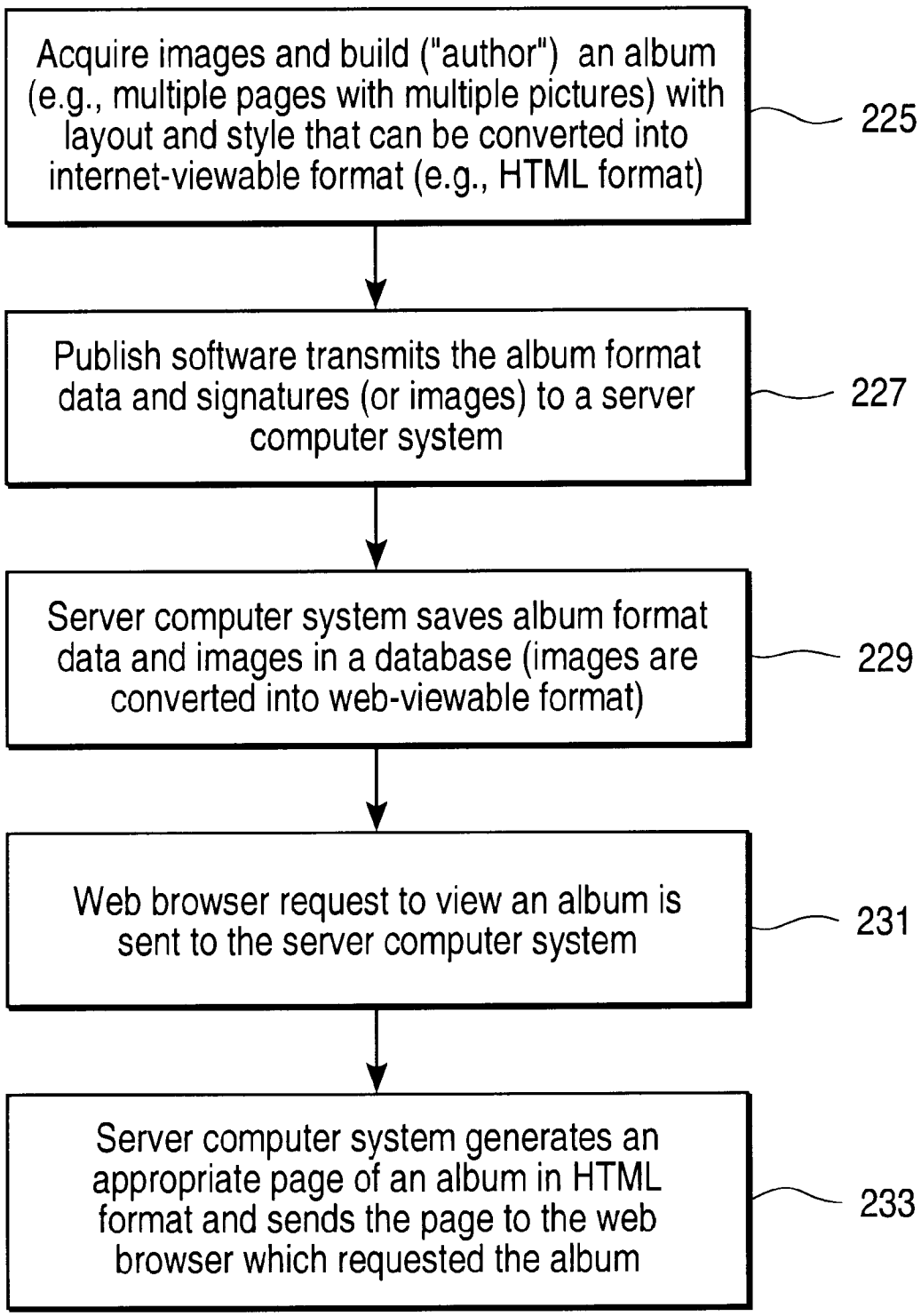

Acquire images and build ("author") an album (e.g., multiple pages with multiple pictures) with layout and style that can be converted into internet-viewable format (e.g., HTML format)
225
Publish software transmits the album format data and signatures (or images) to a server computer system
227
Server computer system saves album format data and images in a database (images are converted into web-viewable format)
229
Web browser request to view an album is sent to the server computer system
231
Server computer system generates an appropriate page of an album in HTML format and sends the page to the web browser which requested the album
233

FIG. 5

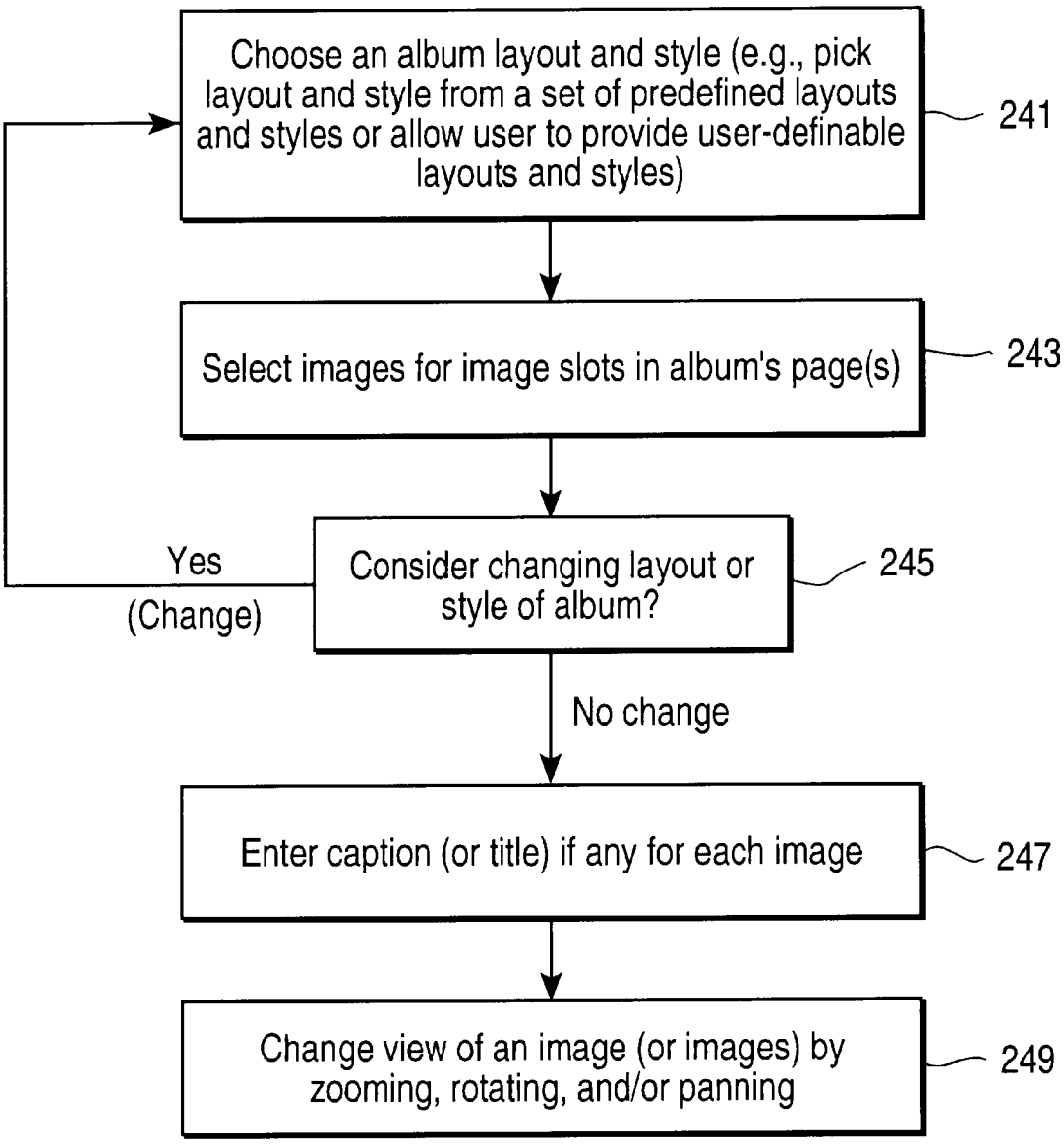

Choose an album layout and style (e.g., pick layout and style from a set of predefined layouts and styles or allow user to provide user-definable layouts and styles)
241
Select images for image slots in album's page(s)
243
Consider changing layout or style of album?
245
Yes
(Change)
No change
Enter caption (or title) if any for each image
247
Change view of an image (or images) by zooming, rotating, and/or panning
249

FIG. 6A

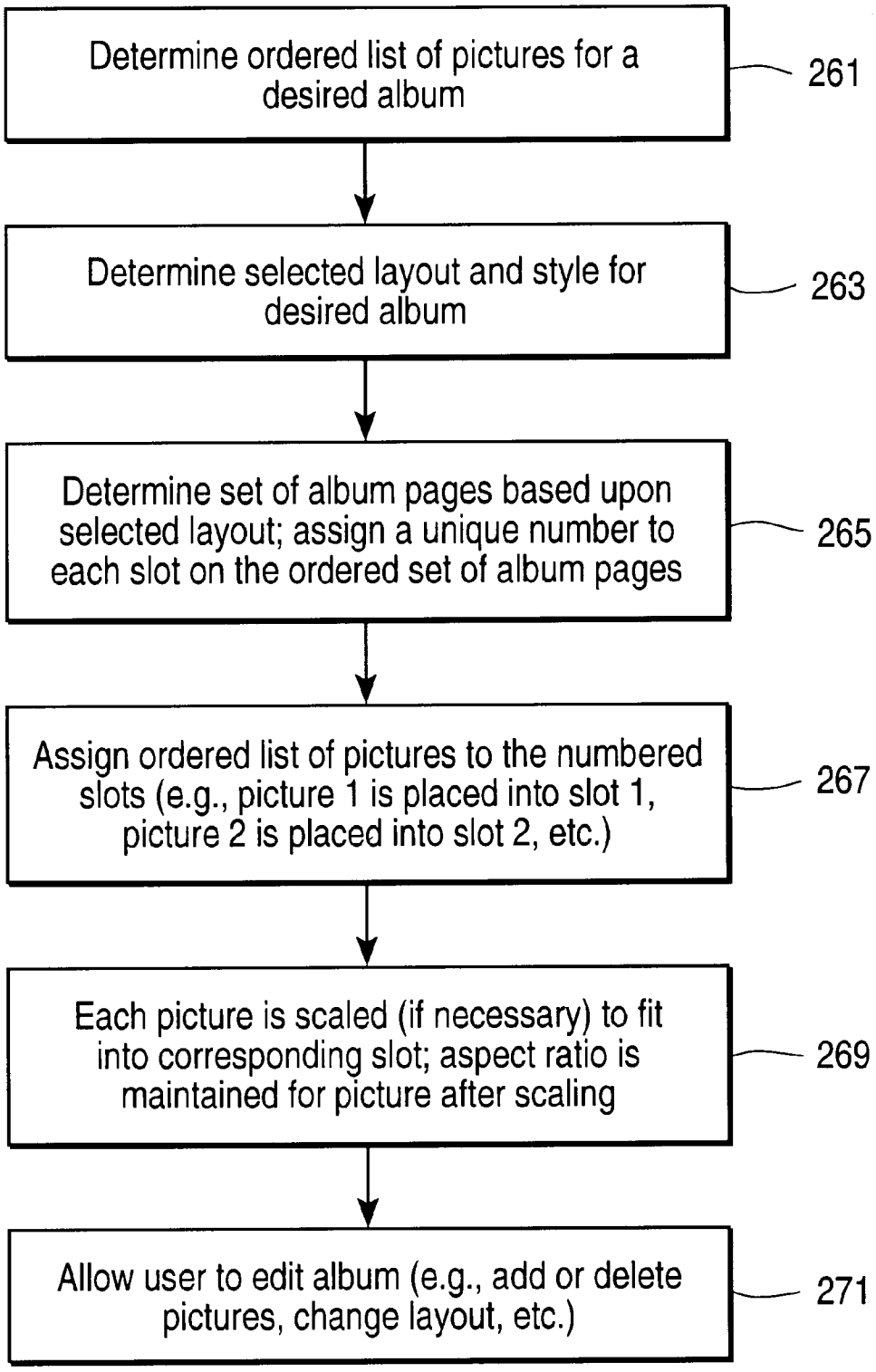

Determine ordered list of pictures for a desired album
261
Determine selected layout and style for desired album
263
Determine set of album pages based upon selected layout; assign a unique number to each slot on the ordered set of album pages
265
Assign ordered list of pictures to the numbered slots (e.g., picture 1 is placed into slot 1, picture 2 is placed into slot 2, etc.)
267
Each picture is scaled (if necessary) to fit into corresponding slot; aspect ratio is maintained for picture after scaling
269
Allow user to edit album (e.g., add or delete pictures, change layout, etc.)
271

FIG. 6B

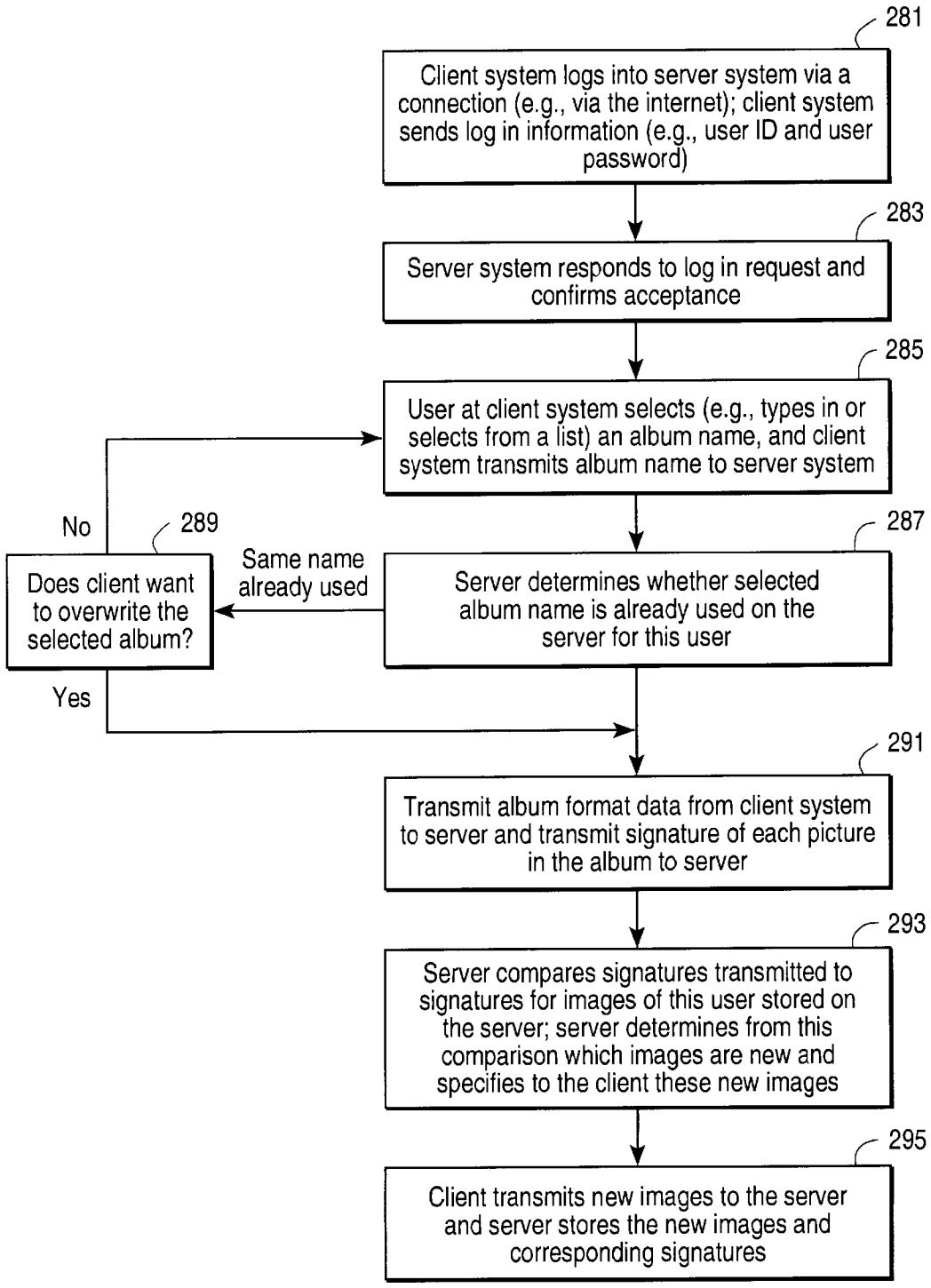

281
Client system logs into server system via a connection (e.g., via the internet); client system sends log in information (e.g., user ID and user password)
283
Server system responds to log in request and confirms acceptance
285
User at client system selects (e.g., types in or selects from a list) an album name, and client system transmits album name to server system
287
Server determines whether selected album name is already used on the server for this user
Same name already used
289
Does client want to overwrite the selected album?
No
Yes
291
Transmit album format data from client system to server and transmit signature of each picture in the album to server
293
Server compares signatures transmitted to signatures for images of this user stored on the server; server determines from this comparison which images are new and specifies to the client these new images
295
Client transmits new images to the server and server stores the new images and corresponding signatures

FIG. 7

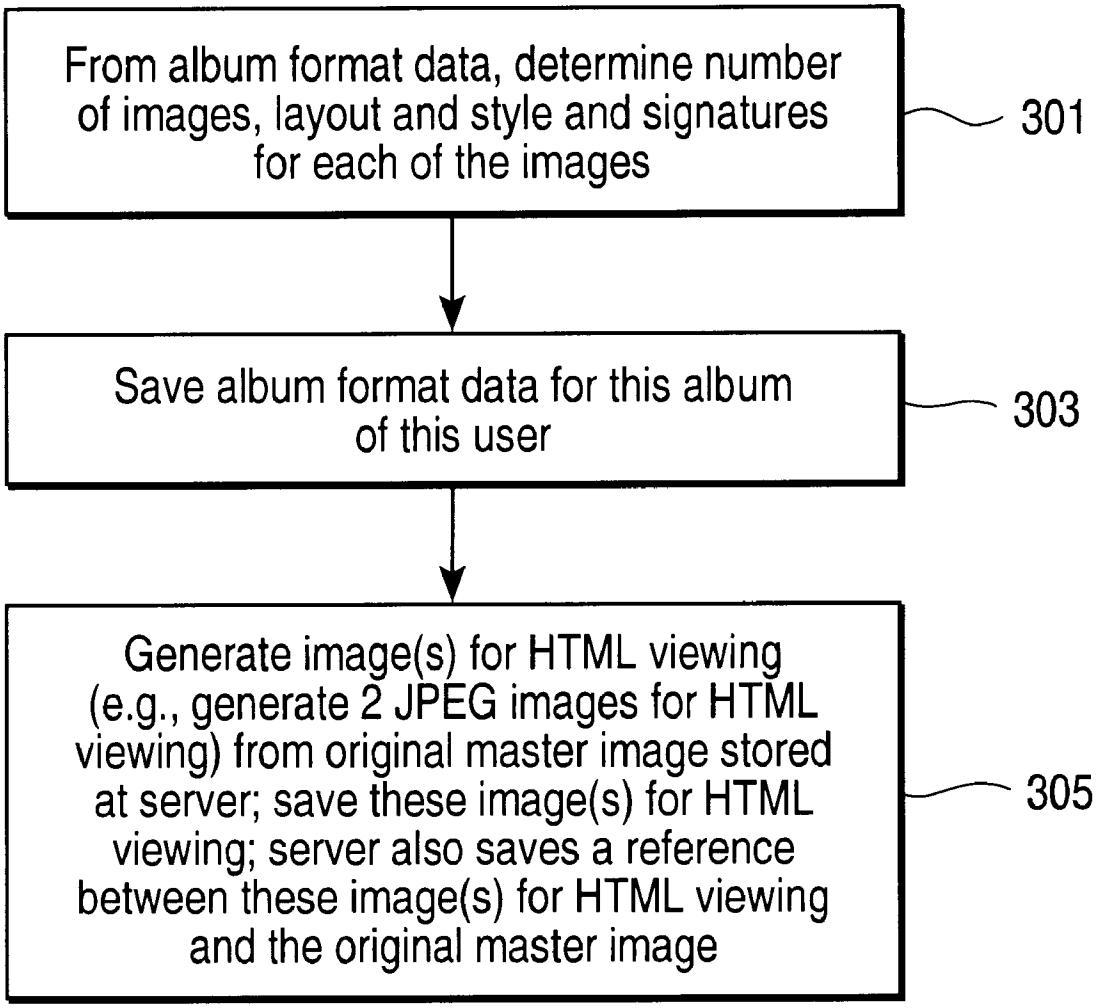

From album format data, determine number of images, layout and style and signatures for each of the images
301
Save album format data for this album of this user
303
Generate image(s) for HTML viewing (e.g., generate 2 JPEG images for HTML viewing) from original master image stored at server; save these image(s) for HTML viewing; server also saves a reference between these image(s) for HTML viewing and the original master image
305

FIG. 8

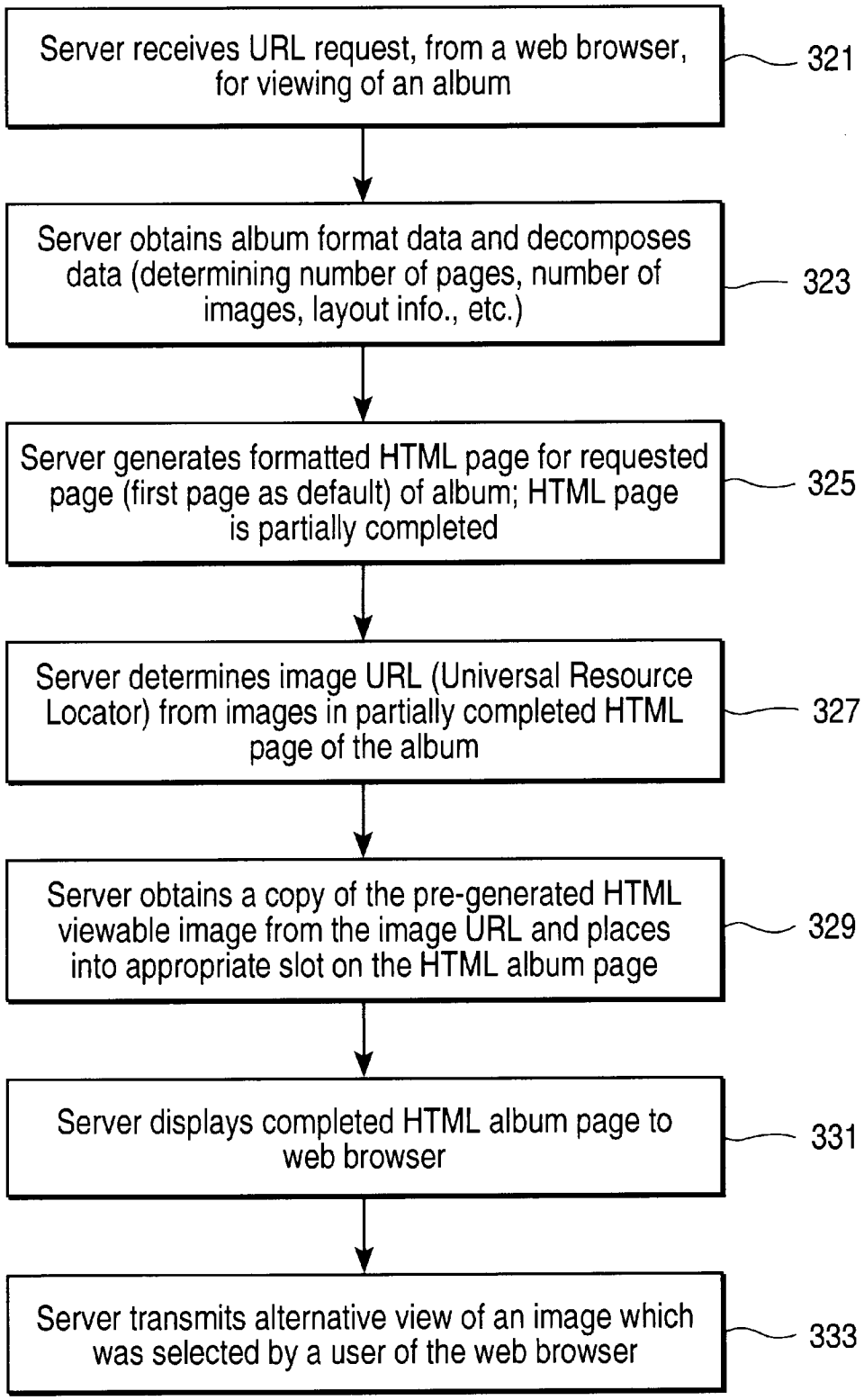

Server receives URL request, from a web browser, for viewing of an album
321
Server obtains album format data and decomposes data (determining number of pages, number of images, layout info., etc.)
323
Server generates formatted HTML page for requested page (first page as default) of album; HTML page is partially completed
325
Server determines image URL (Universal Resource Locator) from images in partially completed HTML page of the album
327
Server obtains a copy of the pre-generated HTML viewable image from the image URL and places into appropriate slot on the HTML album page
329
Server displays completed HTML album page to web browser
331
Server transmits alternative view of an image which was selected by a user of the web browser
333

FIG. 9

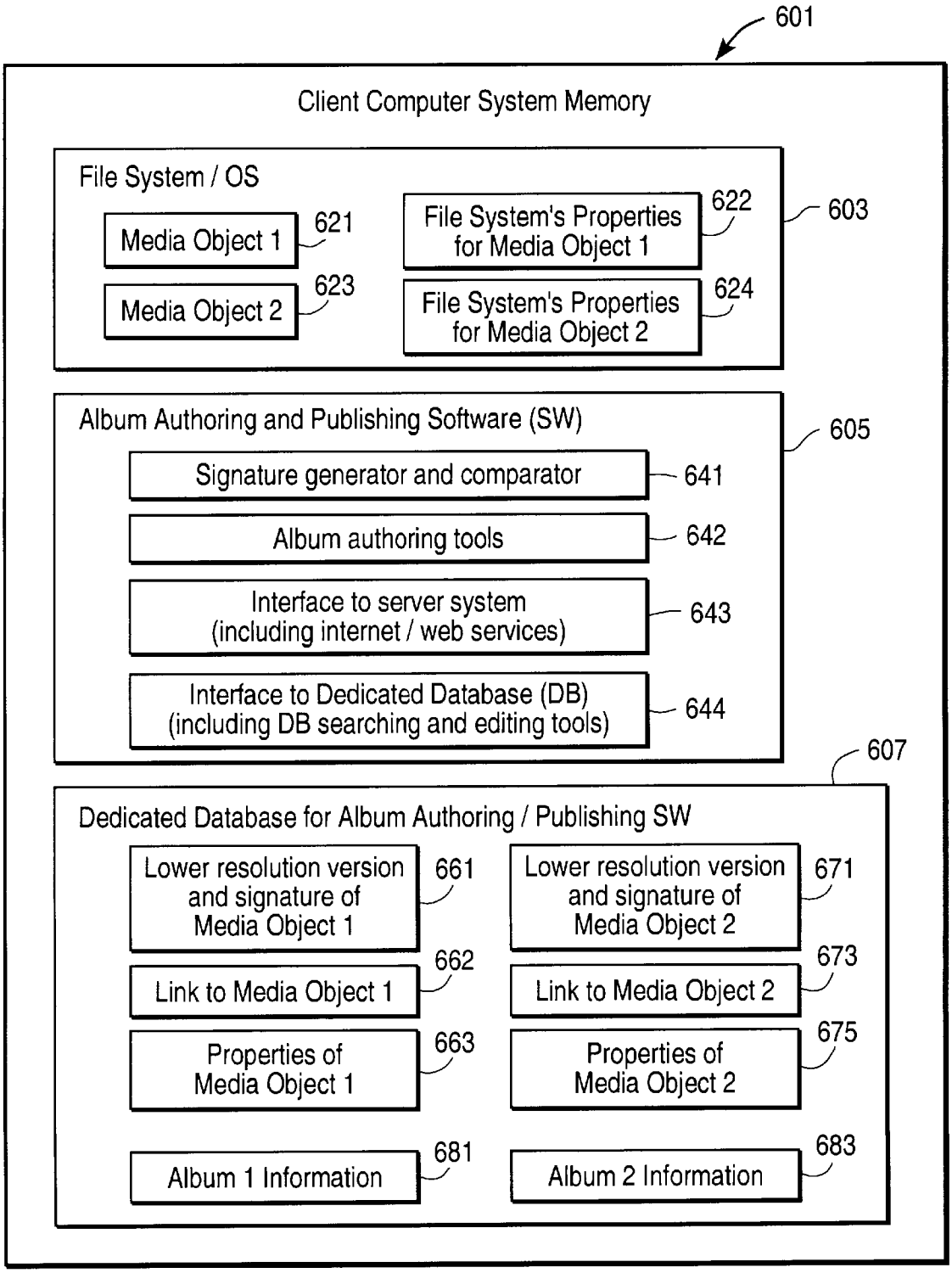

601
Client Computer System Memory
File System / OS
Media Object 1
621
Media Object 2
623
File System's Properties for Media Object 1
622
File System's Properties for Media Object 2
624
603
Album Authoring and Publishing Software (SW)
605
Signature generator and comparator
641
Album authoring tools
642
Interface to server system (including internet / web services)
643
Interface to Dedicated Database (DB) (including DB searching and editing tools)
644
607
Dedicated Database for Album Authoring / Publishing SW
Lower resolution version and signature of Media Object 1
661
Link to Media Object 1
662
Properties of Media Object 1
663
Album 1 Information
681
Lower resolution version and signature of Media Object 2
671
Link to Media Object 2
673
Properties of Media Object 2
675
Album 2 Information
683

FIG. 10

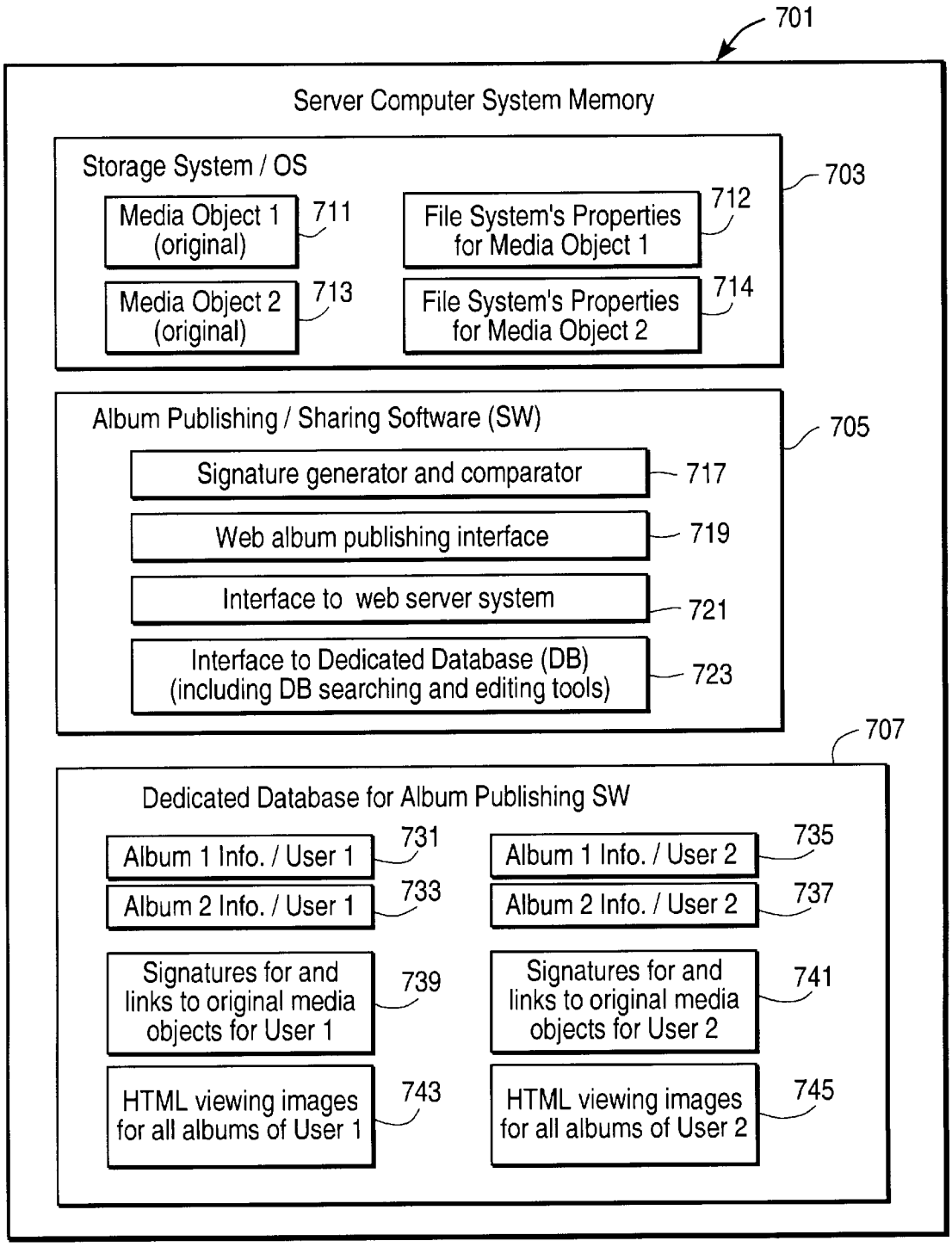

701
Server Computer System Memory
Storage System / OS
703
Media Object 1 (original)
711
File System's Properties for Media Object 1
712
Media Object 2 (original)
713
File System's Properties for Media Object 2
714
Album Publishing / Sharing Software (SW)
705
Signature generator and comparator
717
Web album publishing interface
719
Interface to web server system
721
Interface to Dedicated Database (DB) (including DB searching and editing tools)
723
707
Dedicated Database for Album Publishing SW
Album 1 Info. / User 1
731
Album 1 Info. / User 2
735
Album 2 Info. / User 1
733
Album 2 Info. / User 2
737
Signatures for and links to original media objects for User 1
739
Signatures for and links to original media objects for User 2
741
HTML viewing images for all albums of User 1
743
HTML viewing images for all albums of User 2
745

FIG. 11

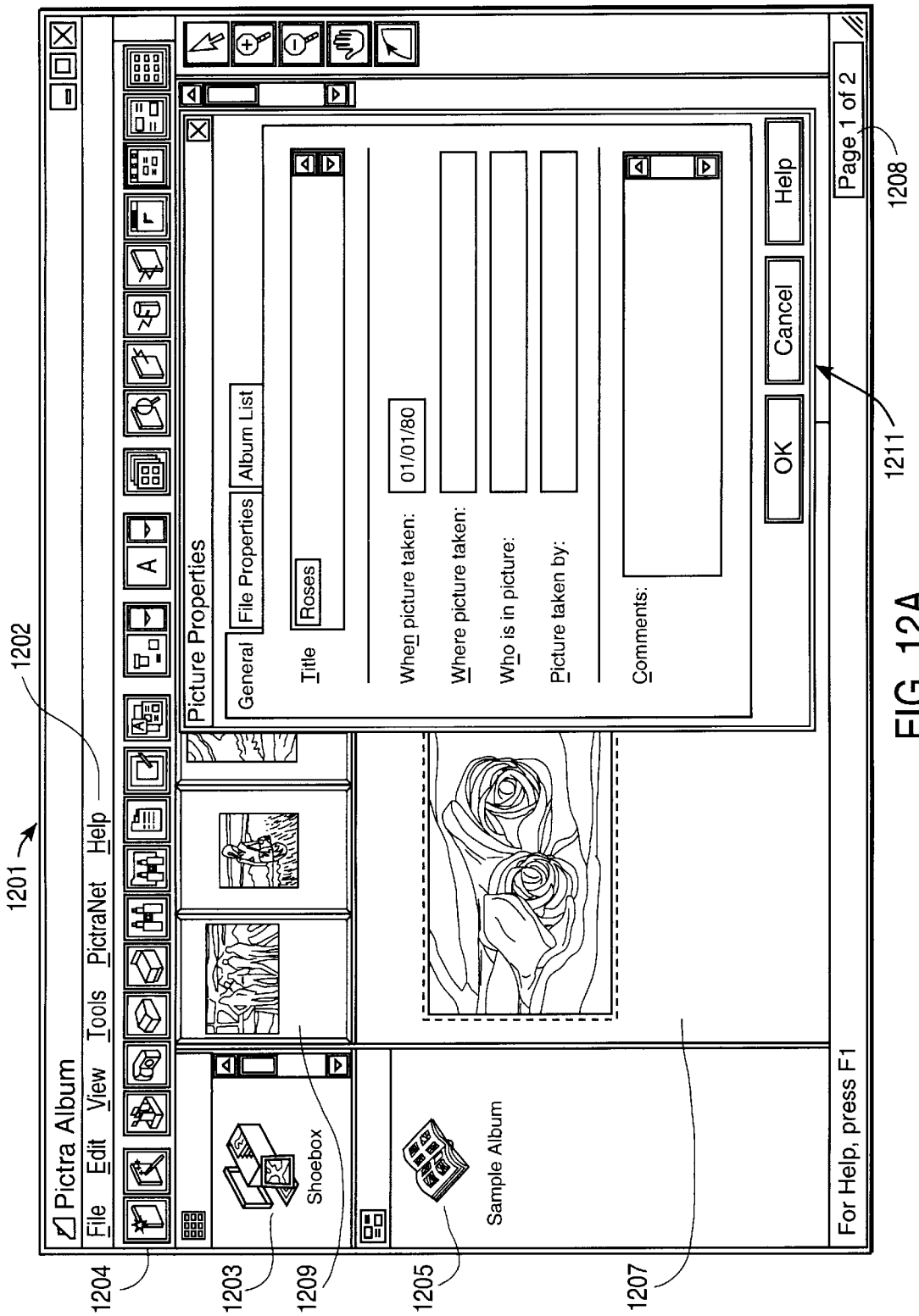

1201
1202
Pictra Album
File Edit View Tools PictraNet Help
Picture Properties
General
File Properties
Album List
Title
Roses
When picture taken:
01/01/80
Where picture taken:
Who is in picture:
Picture taken by:
Comments:
OK
Cancel
Help
Page 1 of 2
1208
1211
Shoebox
Sample Album
For Help, press F1
1204
1203
1209
1205
1207

FIG. 12A

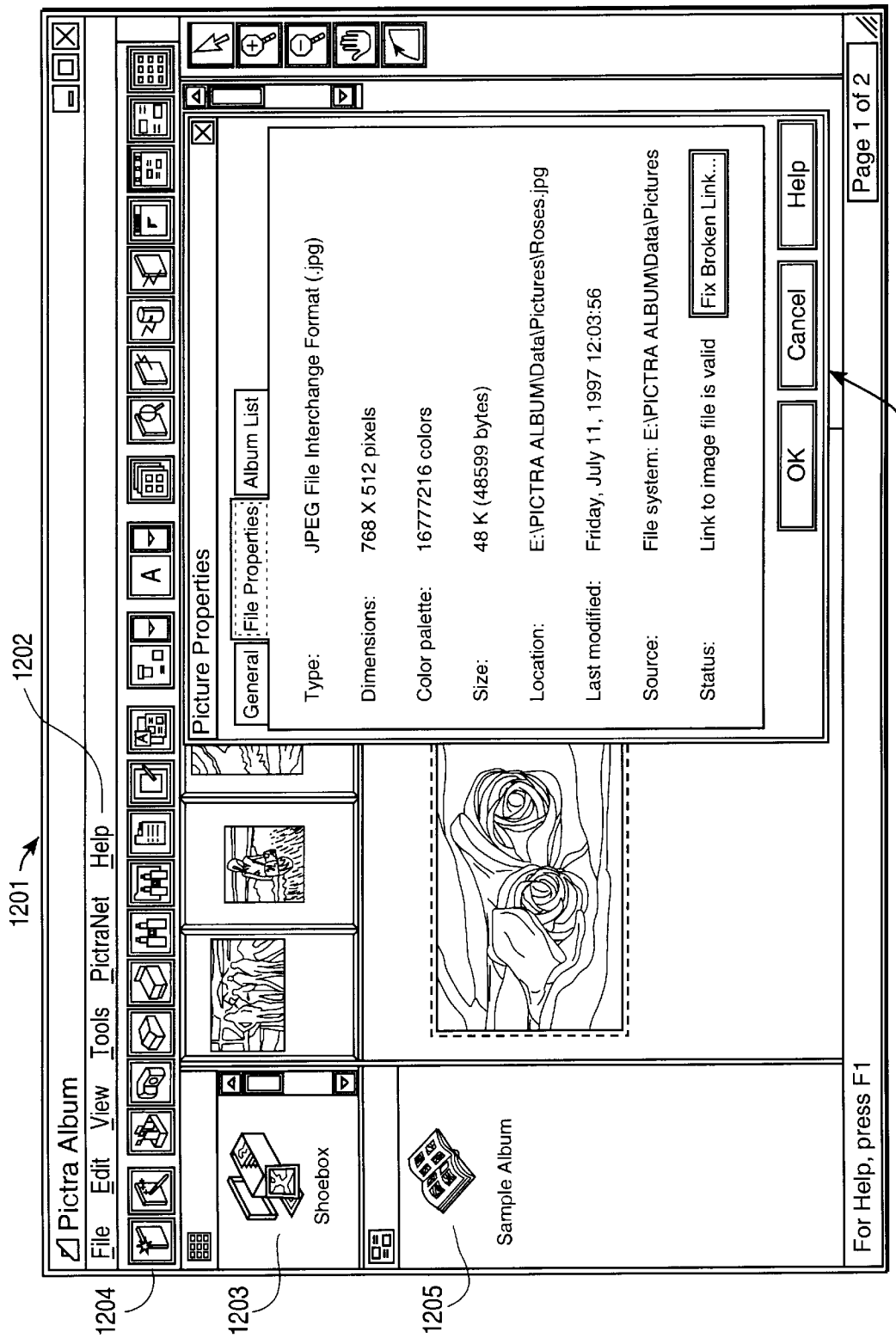

1201
1202
Pictra Album
File Edit View Tools PictraNet Help
Picture Properties
General File Properties Album List
Type: JPEG File Interchange Format (.jpg)
Dimensions: 768 X 512 pixels
Color palette: 16777216 colors
Size: 48 K (48599 bytes)
Location: E:\PICTRA ALBUM\Data\Pictures\Roses.jpg
Last modified: Friday, July 11, 1997 12:03:56
Source: File system: E:\PICTRA ALBUM\Data\Pictures
Status: Link to image file is valid
Fix Broken Link...
OK
Cancel
Help
Page 1 of 2
1211a
Shoebox
Sample Album
1204
1203
1205
For Help, press F1

FIG. 12B

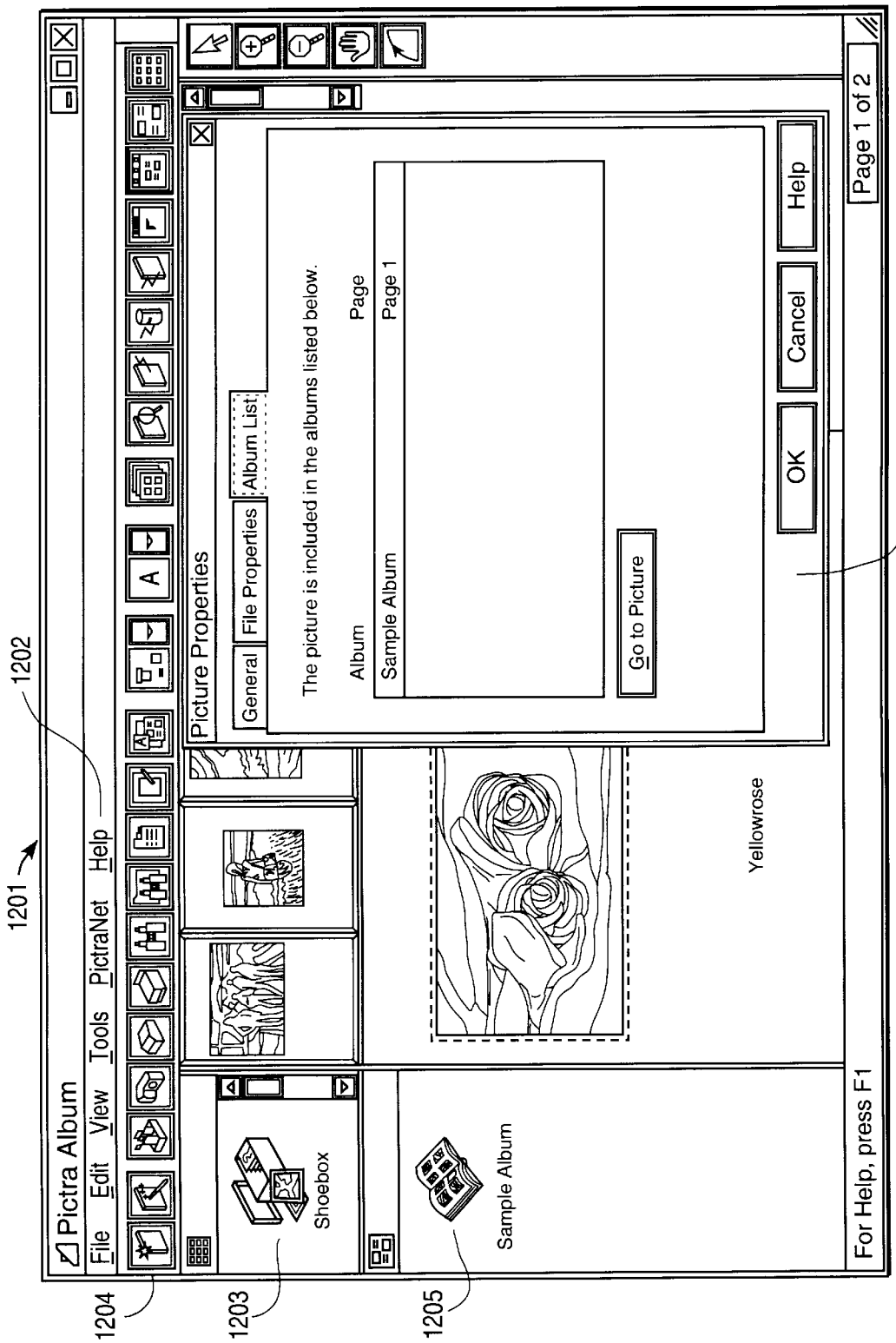

1201
1202
Pictra Album
File Edit View Tools PictraNet Help
Picture Properties
General
File Properties
Album List
The picture is included in the albums listed below.
Album
Page
Sample Album
Page 1
Go to Picture
OK
Cancel
Help
Page 1 of 2
1211b
Yellowrose
Shoebox
Sample Album
For Help, press F1
1204
1203
1205

FIG. 12C

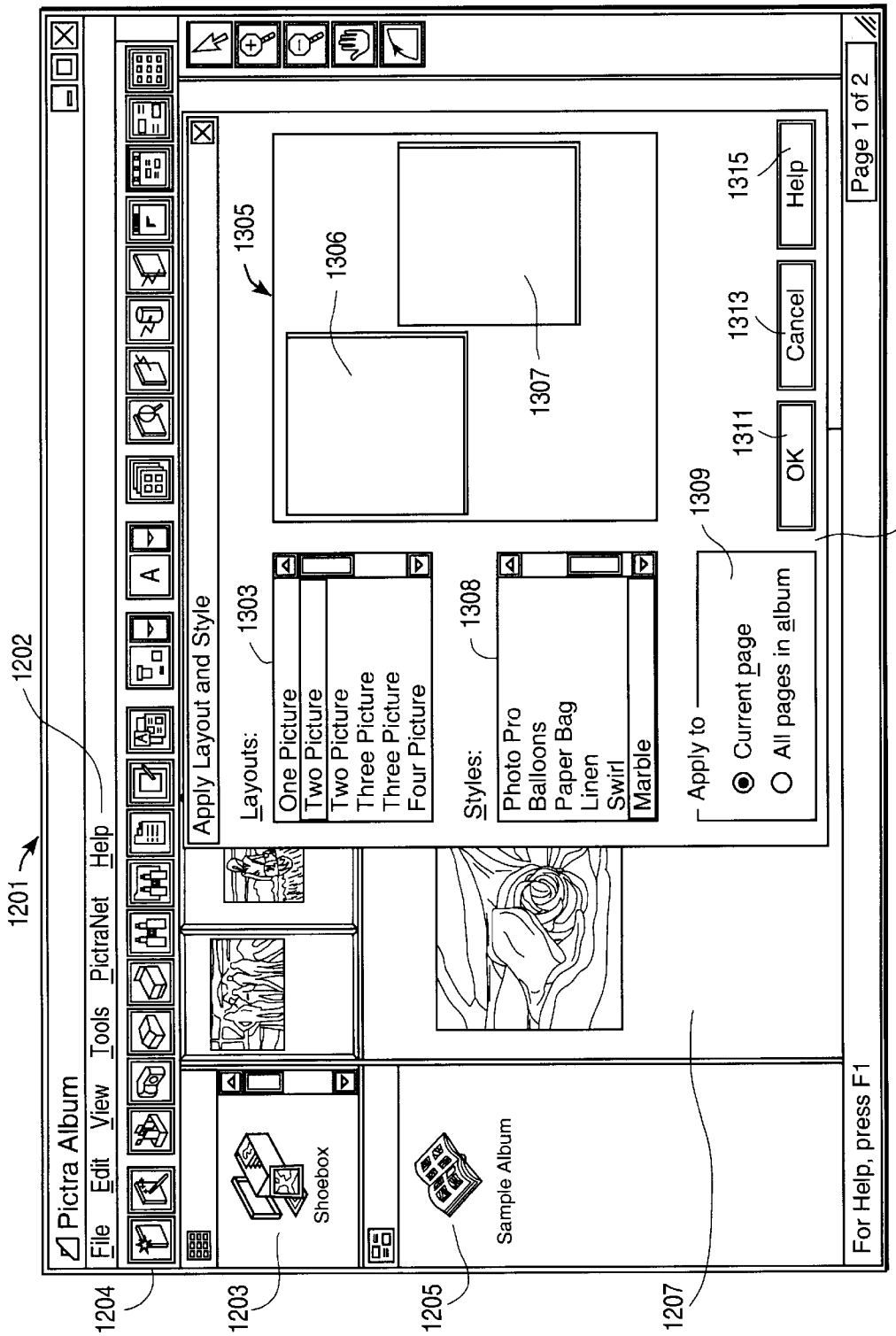

Pictra Album
File Edit View Tools PictraNet Help
Apply Layout and Style
Layouts:
One Picture
Two Picture
Two Picture
Three Picture
Three Picture
Four Picture
Styles:
Photo Pro
Balloons
Paper Bag
Linen
Swirl
Marble
Apply to
Current page
All pages in album
OK
Cancel
Help
Page 1 of 2
Shoebox
Sample Album
For Help, press F1
1201
1202
1203
1204
1205
1207
1301
1303
1305
1306
1307
1308
1309
1311
1313
1315

FIG. 13

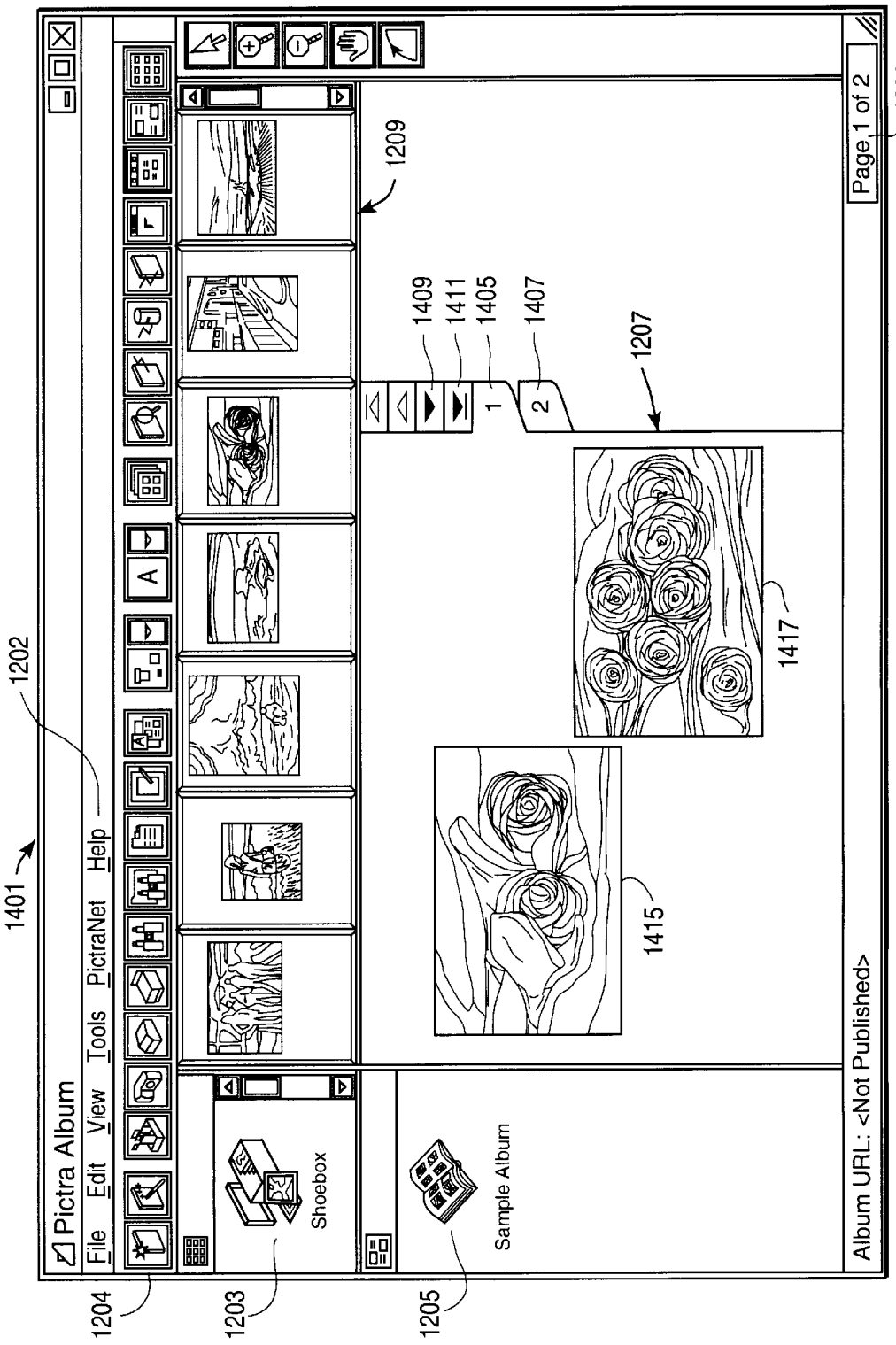

1401
1202
Pictra Album
File Edit View Tools PictraNet Help
1204
1203
Shoebox
1205
Sample Album
1209
1409
1411
1405
1407
1207
1208
Page 1 of 2
1417
1415
Album URL: <Not Published>

FIG. 14A

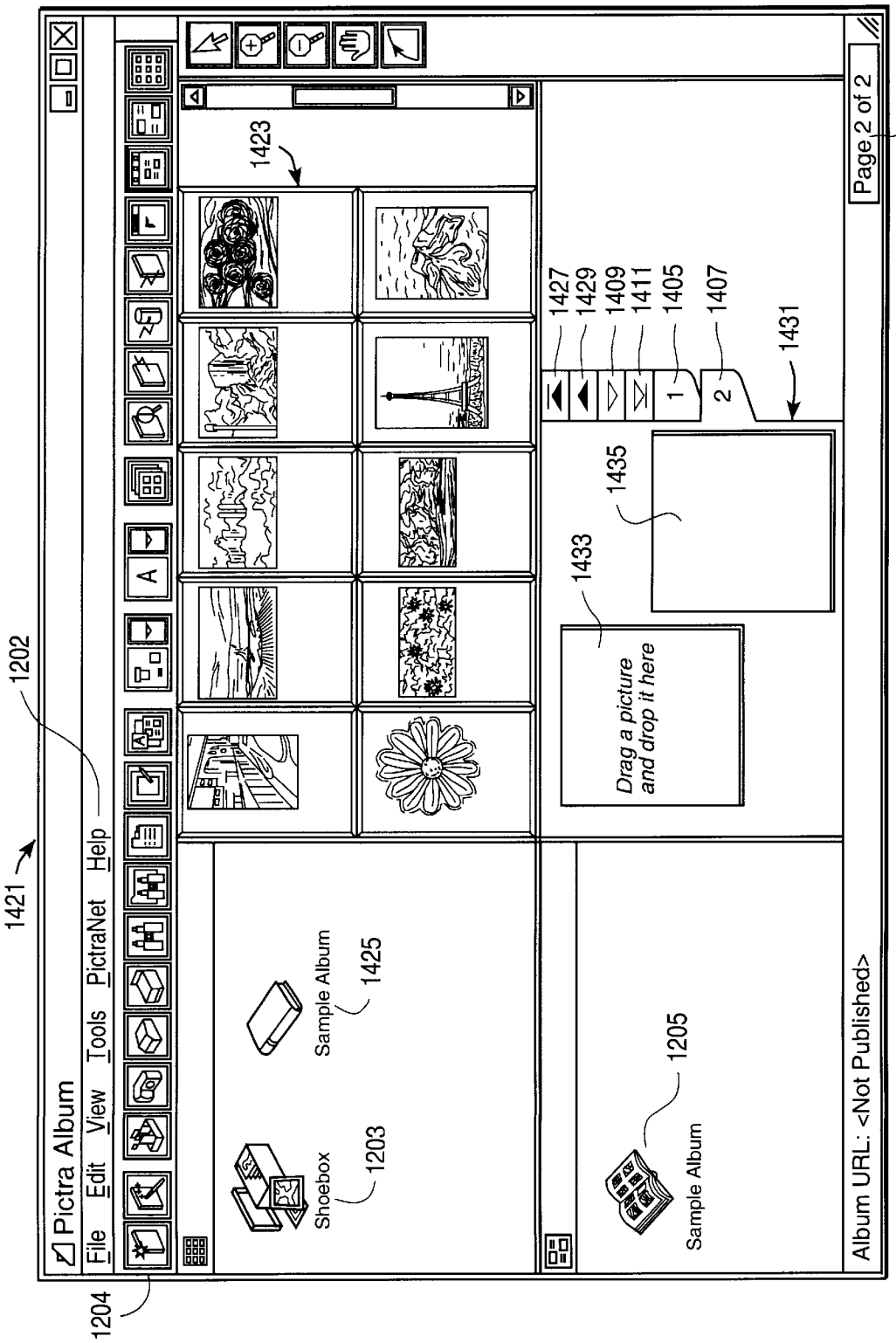

1421
1202
Pictra Album
File Edit View Tools PictraNet Help
1423
1427
1429
1409
1411
1405
1407
1431
1208
Page 2 of 2
1435
1433
Drag a picture and drop it here
Sample Album
1425
Shoebox
1203
1205
Sample Album
Album URL: <Not Published>
1204

FIG. 14B

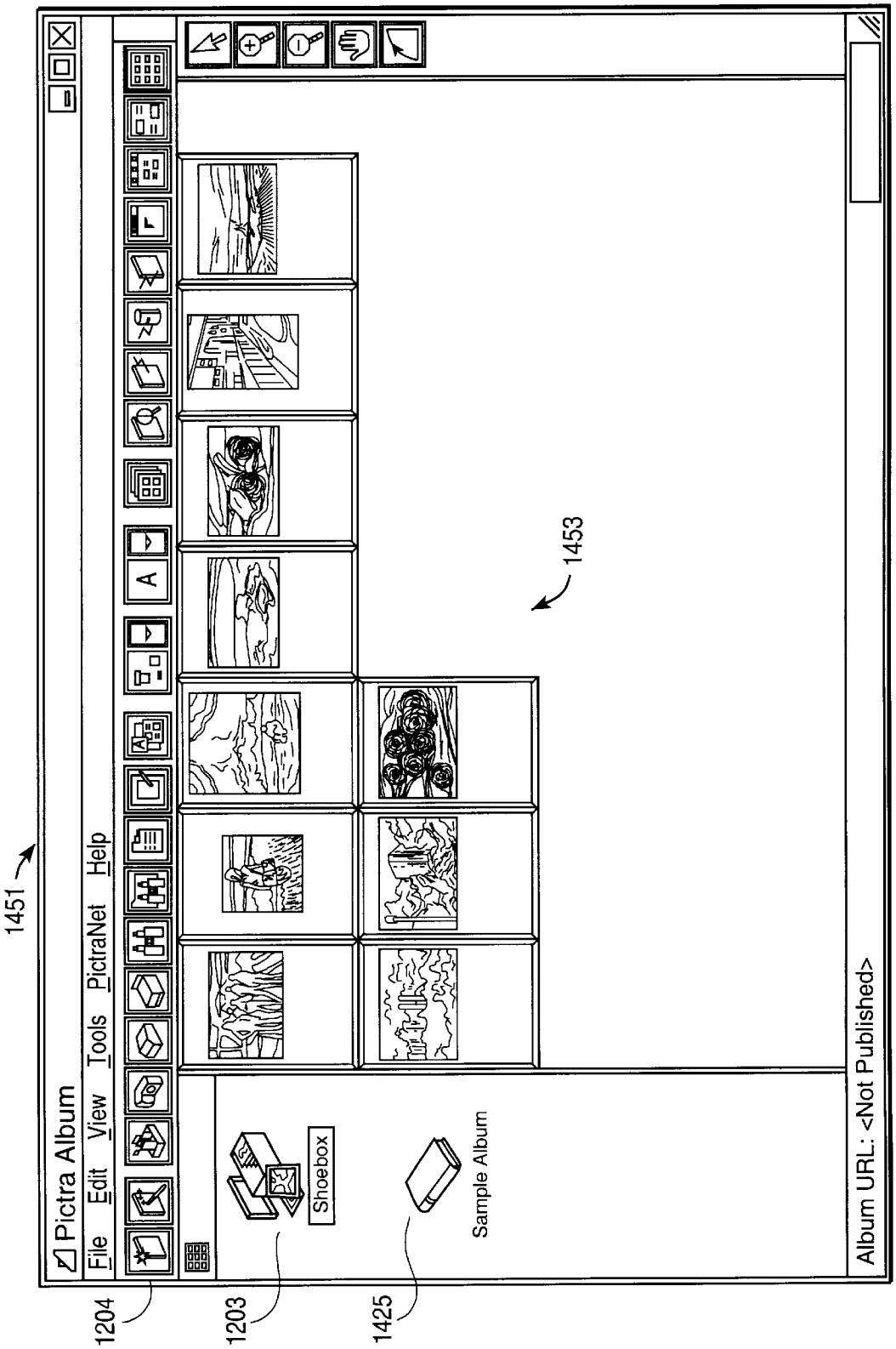

1451
Pictra Album
File Edit View Tools PictraNet Help
1453
Shoebox
Sample Album
Album URL: <Not Published>
1204
1203
1425

FIG. 14C

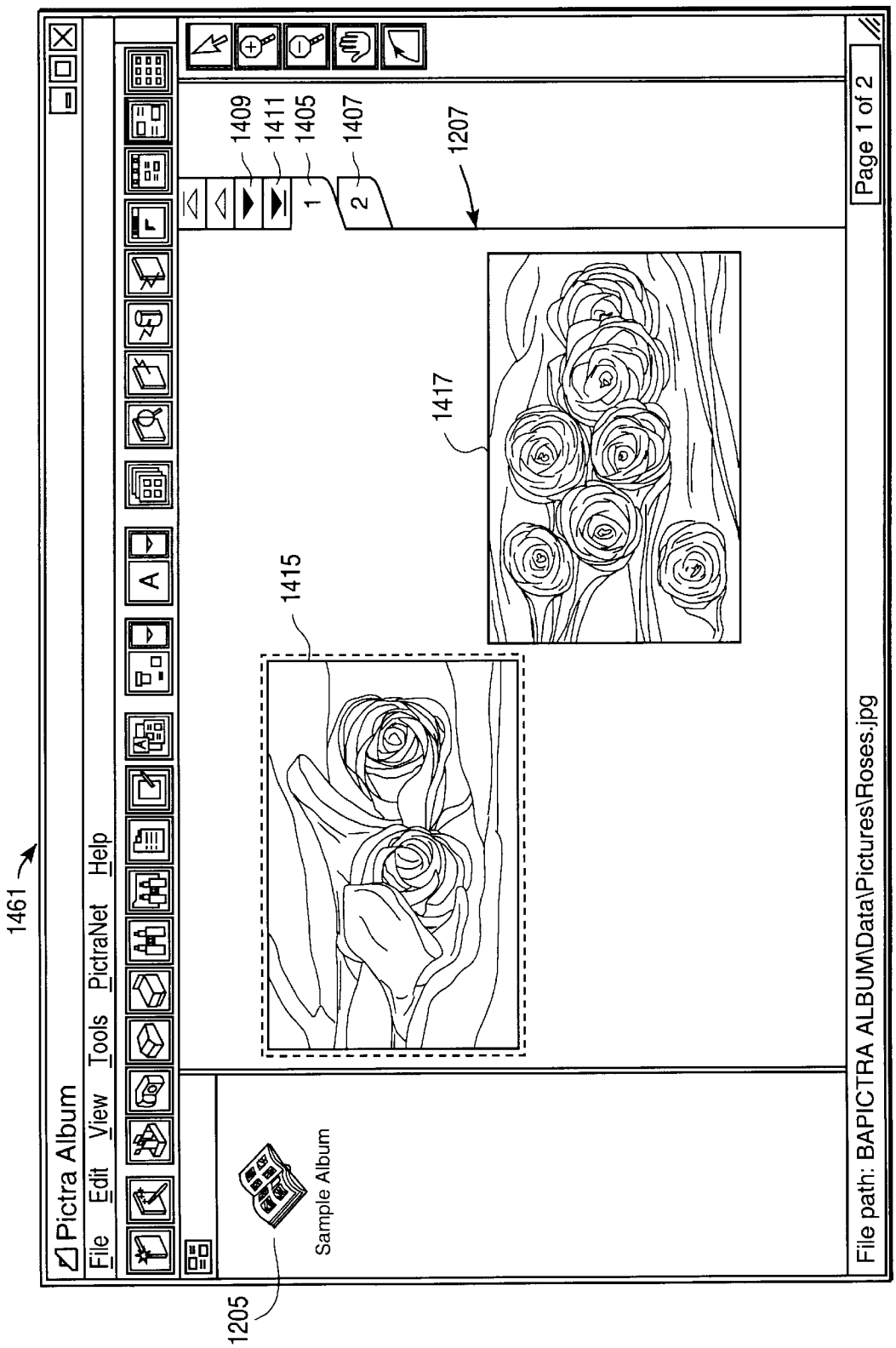

1461
Pictra Album
File Edit View Tools PictraNet Help
1409
1411
1405
1407
1207
1
2
Page 1 of 2
1417
1415
Sample Album
1205
File path: BAPICTRA ALBUM\Data\Pictures\Roses.jpg

FIG. 14D

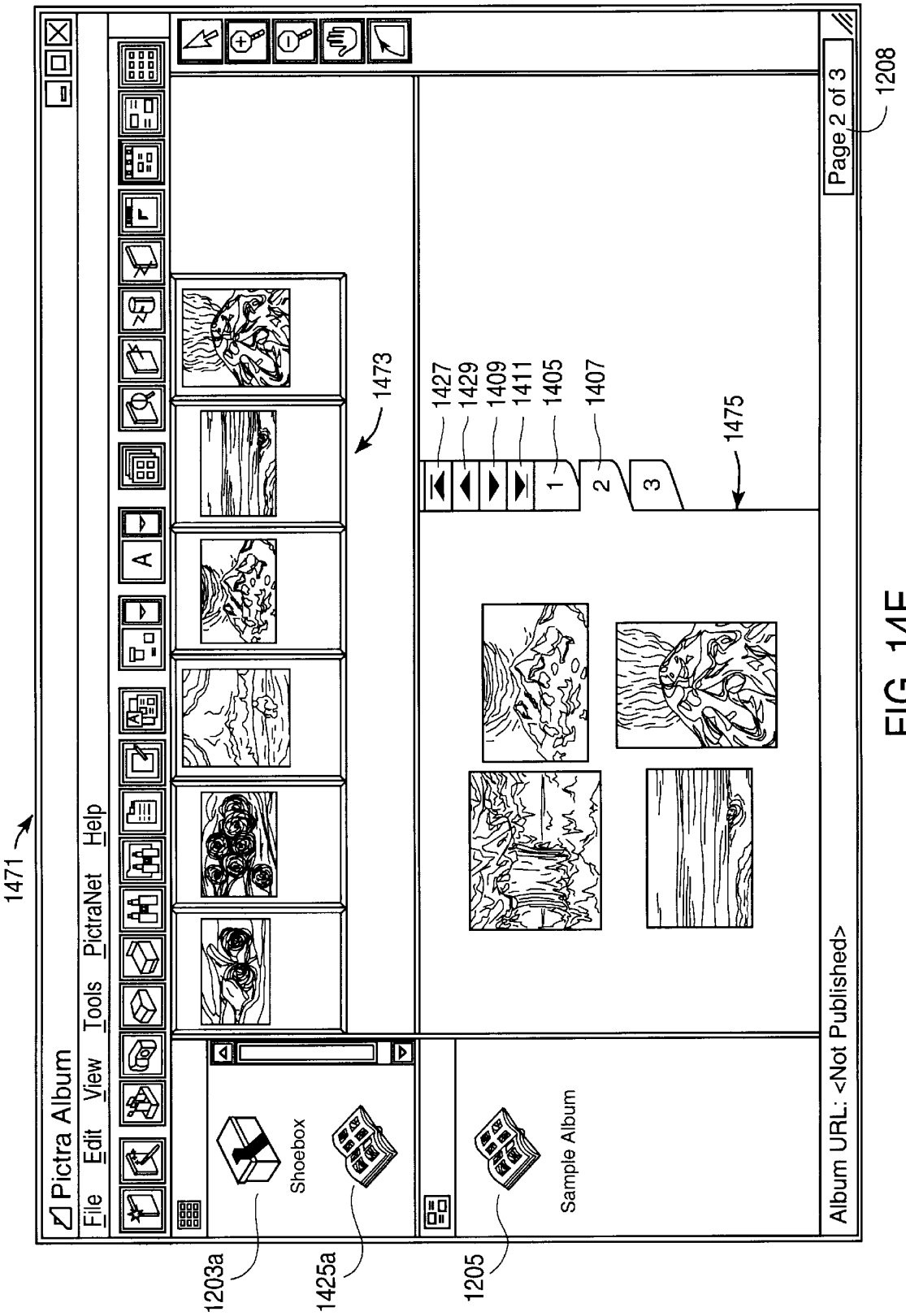

1471
Pictra Album
File Edit View Tools PictraNet Help
1473
1427
1429
1409
1411
1405
1407
1475
1
2
3
Page 2 of 3
1208
Shoebox
Sample Album
Album URL: <Not Published>
1203a
1425a
1205

FIG. 14E

METHODS AND APPARATUSES FOR DISTRIBUTING A COLLECTION OF DIGITAL MEDIA OVER A NETWORK WITH AUTOMATIC GENERATION OF PRESENTABLE MEDIA

FIELD OF THE INVENTION

The present invention relates to methods and apparatuses for distributing digital media over a network. Particularly, the present invention relates to methods and apparatuses for publishing digital images over a network operating according to Internet or intranet protocols.

BACKGROUND OF THE INVENTION

Digital processing systems, such as conventional computer systems, often can accept input from a digital acquisition device, such as a digital camera or a scanner. Examples of digital cameras and scanners are well known in the art. These devices may be used to capture an image, such as a picture of a person, and then the image may be input into a computer system. For example, the output of a digital camera may be input to a computer system which is executing a digital photography software program, such as Photoshop from Adobe Systems, Inc. of San Jose, Calif. Once the digital image is inputted into the computer system, it is stored on a computer readable storage medium, such as a hard disk, a floppy disk, an optical disk, or other well known computer readable storage media. The storage of the media on the computer readable storage media is controlled by a file management system which is usually considered to be a disk operating system software which is also running on the computer system. Thus, the result of the conventional digital photography software is the storage of a file which contains the content of the digital picture on a computer readable medium which is controlled by the file management system of the computer system.

The user can then modify the image, print the image, and perform other operations with the image. One such operation is the distribution or publication of the image over a network, such as the World Wide Web or the Internet.

FIG. **1** shows a process in the prior art which typically requires at least two software programs at the computer sy tem which first acquires the digital image. In particular, a digital photography computer program is typically required to capture the image from a digital camera or from a scanner, and a separate web authoring computer program, such as Front Page '97 from Microsoft Corporation of Redmond, Wash. is also required. In addition, the web authoring software (or some other software) must transfer the files and images to an Internet service provider which then must link the files and images and maintain the HTML (hypertext markup language) formatted documents at a web server.

FIG. **1** shows the various steps required in this complicated procedure in order to make digital images available for viewing over the Internet to web browsers. The method starts by acquiring in step **10** an image from a digital camera. This acquisition typically occurs by a digital photography program, such as Adobe's Photoshop, which receives the input from the digital camera and causes the inputted image to be saved in the file management system of the computer system. Then in step **12**, a separate computer program, such as a web authoring software program creates HTML files. These files may have insertion points for various digital images, such as those acquired from a digital camera in step **10**. Then in step **14**, the web authoring software, such as Microsoft's Front Page '97, transfers the HTML files and also separately transfers the digital images stored in step **10** to an Internet service provider. Typically the files are transferred according to the FTP protocol. In step **16**, the Internet service provider links the HTML files and the appropriate images and maintains persistently the HTML pages from the files and images on a web server, and this allows web browsers to view the HTML pages from the web server provided by the Internet service provider (ISP)

The foregoing procedure is relatively complicated, requires the user of the digital camera to have some familiarity with HTML document creation, and also requires the use of at least two separate programs on of the computer system which receives the input from the digital camera. Furthermore, the Internet service provider must perform the linking operation in order to maintain HTML pages which present the images to web browsers over the Internet.

While it is possible to use e-mail computer programs or public bulletin board services to attempt to distribute digital media over a network, such programs or systems are not designed to provide for the generation of a collection of digital media and then the transmission of the collection and then the automatic generation from the collection of viewable media. Also, certain organizations, such as real estate realtors have attempted to distribute digital media over the Internet. However, this distribution is believed to be as complicated as the method shown in FIG. **1**.

Given the complexity of the foregoing tasks, it is desirable to allow a user of a digital camera to easily distribute or publish images from the digital camera or other digital acquisition devices over a network, such as the Internet.

SUMMARY OF THE INVENTION

The present invention discloses methods and apparatuses for distributing a collection of digital media on a network. A method in one example of the invention generates a collection of digital media at a client digital processing system. Then collection information is transmitted from the client digital processing system to a server digital processing system. The collection information describes the collection of digital media. A plurality of presentable media is then automatically generated from the collection information. Each of the plurality of presentable media is capable of being presented to other client digital processing systems which are coupled to a network. Typically this network is operating according to the hypertext transfer protocol (HTTP) and both the client and the server system are coupled to this network. In one particular example of the present invention, the digital media includes digital images and the presentable media includes viewable images. Further, the client digital processing system and the server digital processing system are each programmed to interact together such that the server digital processing system generates automatically, from the collection information, the plurality of viewable images.

Computer systems which practice the methods of the invention are also described. Further, computer readable media having software which allows the computer systems to perform the methods of the present invention are described.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** shows a method according to the prior art of distributing a digital image from a digital camera such that users of web browsers may view the image.

FIG. **2** shows an example of several client computer systems coupled to the Internet and a server computer

system with a picture database or a database of other digital media according to one embodiment of the present invention.

FIG. **3** shows an example of a computer system which may be used with the present invention.

FIG. **4** shows a method for using a digital acquisition device to load pictures into a computer system according to one aspect of the present invention.

FIG. **5** shows an overall method for automatically generating viewable images over a network according to one aspect of the present invention.

FIG. **6**A is a flowchart illustrating one method for creating a picture album according to one aspect of the present invention; the method shown in FIG. **6**A is in one embodiment a more detailed representation of the steps performed in step **225** of FIG. **5**.

FIG. **6**B is a flowchart illustrating one example of a method of creating a picture album according to one aspect of the present invention.

FIG. **7** shows a flowchart which illustrates various steps which may be performed as part of the step **227** shown in FIG. **5**.

FIG. **8** shows a flowchart which illustrates several steps in one embodiment which may be performed as part of the step **229** of FIG. **5**.

FIG. **9** shows one method according to the present invention for performing the steps of step **233** shown in FIG. **5**.

FIG. **10** shows an example of a computer readable storage medium for a client computer system which may be used with one aspect of the present invention.

FIG. **11** illustrates a computer readable storage medium for a server computer system which may be used with one aspect of the present invention.

FIG. **12**A, illustrate a particular graphical user interface which depicts an example of the various information which may be maintained in a picture database according to one aspect of the present invention.

FIG. **12**B illustrates a particular graphical user interface which depicts an example of the various information which may be maintained in a picture database according to one aspect of the present invention.

FIG. **12**C illustrates a particular graphical user interface which depicts an example of the various information which may be maintained in a picture database according to one aspect of the present invention.

FIG. **13** shows a graphical user interface for allowing a user to select a layout and a style for a picture album according to one aspect of the present invention.

FIG. **14**A illustrates one example of a graphical user interface of the present invention which allows the user to view and control various aspects of one or more albums and the pictures in the albums.

FIG. **14**B illustrates one example of a graphical user interface of the present invention which allows the user to view and control various aspects of one or more albums and the pictures in the albums.

FIG. **14**C illustrates one example of graphical user interface of the present invention which allows the user to view and control various aspects of one or more albums and the pictures in the albums.

FIG. **14**D illustrates one example of graphical user interface of the present invention which allows the user to view and control various aspects of one or more albums and the pictures in the albums.

FIG. **14**E illustrates one example of graphical user interface of the present invention which allows the user to view and control various aspects of one or more albums and the pictures in the albums.

DETAILED DESCRIPTION

The subject invention will be described with reference to numerous details set forth below, and the accompanying drawings will illustrate the invention. The following description and the drawings are illustrative of the invention and are not to be construed as limiting the invention. Numerous specific details are described to provide a thorough understanding of the present invention. However, in certain instances, well known or conventional details are not described in order to not unnecessarily obscure the present invention in detail. In the drawings, the same element is labeled with the same reference numeral.

FIG. **2** shows several computer systems which are coupled together through the Internet **103**. It will be appreciated herein that the term "Internet" refers to a network of networks which uses certain protocols (e.g. the TCP/IP protocol, and possibly other protocols such as the HTTP (hypertext transfer protocol) for HTML (hypertext markup language) documents). The physical connections of the Internet and the protocols and communication procedures of the Internet are well known to those in the art. Access to the Internet **103** is typically provided by Internet service providers (ISP), such as the ISPs **105** and **107**. Users on client systems, such as client computer systems **121**, **125**, **135**, and **137** obtain access to the Internet through the Internet service providers, such as ISPs **105** and **107**. Access to the Internet allows users of the client computer systems to exchange information, receive and send e-mails, and view documents, such as documents which have been prepared in the HTML format. These documents are often provided by web servers, such as web server **109** which is considered to be "on" the Internet. Often these web servers are provided by the ISPs, such as ISP **105**, although a computer system may be set up and connected to the Internet without that system being also an ISP as is well known in the art.

The web server **109** is typically at least one computer system which operates as a server computer system and is configured to operate with the protocols of the World Wide Web (WWW) and is coupled to the Internet. Optionally, the web server **109** may be part of an ISP which provides access to the Internet for client systems. The web server **109** is shown coupled to the server computer system **111** which itself is coupled to a picture database **110**, which may be considered a form of a media database. It will be appreciated that while two computer systems **109** and **111** are shown in FIG. **2**, the web server system **109** and the server computer system **111** may be one computer system having different software components providing the web server functionality and the server functionality provided by the server computer system **111** which will be described further below. Client computer systems **121**, **125**, **135**, and **137** may each, with the appropriate web browsing software, view HTML pages provided by the web server **109**. The ISP **105** provides Internet connectivity to the client computer system **121**

through the modem interface **123** which may be considered part of the client computer system **121**. The client computer system may be a "WINTEL" computer system, a network computer, a Web TV system, or other computer systems. Similarly, the ISP **107** provides Internet connectivity for client systems **125**, **135**, and **137**, although as shown in FIG. **1**, the connections are not the same for these three computer systems. Client computer system **125** is coupled through a modem interface **127** while client computer systems **135** and **137** are part of a local area network (LAN). While FIG. **2** shows the interfaces **123** and **127** as a "modem," it will be appreciated that each of these interfaces may be an analog modem, ISDN modem, cable modem, satellite transmission interface (e.g. "Direct PC"), or other interfaces for coupling a computer system to other computer systems. Client computer systems **135** and **137** are coupled to a LAN bus **133** through network interfaces **139** and **141**, which may be Ethernet network or other network interfaces. The LAN bus is also coupled to a gateway computer system **131** which may provide firewall and other Internet related services for the local area network. This gateway computer system **131** is coupled to the ISP **107** to provide Internet connectivity to the client computer systems **135** and **137**. The gateway computer system **131** may be a conventional server computer system. Also, the web server system **109** may be a conventional server computer system.

FIG. **3** shows one example of a conventional server computer system which may be used as a client computer system or a server computer system or as a web server system. It will also be appreciated that such a computer system may be used to perform many of the functions of an Internet service provider, such as ISP **105**. The computer system **501** interfaces to external systems through the modem or network interface **503**. It will be appreciated that the modem or network interface **503** may be considered to be part of the computer system **501**. This interface **503** may be an analog modem, ISDN modem, cable modem, token ring interface, satellite transmission interface (e.g. "Direct PC"), or other interfaces for coupling a computer system to other computer systems. The computer system **501** includes a processor **505**, which may be a conventional microprocessor such as an Intel Pentium microprocessor or Motorola Power PC microprocessor. Memory **509** is coupled to the processor **505** by a bus **507**. Memory **509** may be dynamic random access memory (DRAM) and may also include static RAM (SRAM). The bus **507** couples the processor **505** to the memory **509** and also to mass memory **515** and to display controller **511** and to the I/O (input/output) controller **517**. The display controller **511** controls in the conventional manner a display on a display device **513** which may be a CRT or liquid crystal display. The input/output devices **519** may include a keyboard, disk drives, printers, a scanner, and other input and output devices, including a mouse or other pointing device. The display controller **511** and the I/O controller **517** may be implemented with conventional well known technology. A digital image input device **521** may be a digital camera which is coupled to an I/O controller **517** in order to allow images from the digital camera to be input into the computer system **501**. The mass memory **515** is often a magnetic hard disk, an optical disk, or another form of storage for large amounts of data. Some of this data is often written, by a direct memory access process, into memory **509** during execution of software in the computer system **501**. It will be appreciated that the computer system **501** is one example of many possible computer systems which have different architectures. For example, WINTEL systems (systems which run a Microsoft Windows operating system on an Intel microprocessor) often have multiple buses, one of which may be considered to be a peripheral bus. Network computers may also be considered to be a computer system which may be used with the present invention. Network computers may not include a hard disk or other mass storage, and the executable programs are loaded from a network connection into the memory **509** for execution by the processor **505**. A Web TV system, which is known in the art, may be considered to be a computer system according to the present invention, but it may not include certain features shown in FIG. **3**, such as certain input or output devices. A typical computer system will usually include at least a processor, memory, and a bus coupling the memory to the processor. It will also be appreciated that the computer system **501** is controlled by operating system software which includes a file management system, such as a disk operating system, which is part of the operating system software. One example of an operating system software with its associated file management system software is the operating system known as Windows '95 from Microsoft Corporation of Redmond, Wash., and its associated file management system, including Windows Explorer. The file management system is typically stored in the mass memory **515** and causes the processor **505** to execute the various steps required by the operating system to input and output data and to store data in memory, including storing files on the mass memory **515**.

FIG. **4** will now be referred to to describe one aspect of the present invention which relates to methods and apparatuses for acquiring a digital image for use in a digital processing system, such as a computer system. The method of FIG. **4** begins in step **201** in which a user inputs digital images from a digital acquisition device, such as a digital camera into a digital processing system. For purposes of explanation, this description often uses the term "digital camera" rather than the term "digital acquisition device." It will be understood that "digital camera" is being used as a shorthand phrase to refer to the general group of digital acquisition devices, and that a digital camera is an example of a digital acquisition device. In step **203**, the user makes selections in a file saving dialog box presented to the user on a display of the computer system in order to store the original images to the file storage device, such as a hard disk. The storage occurs under the control of a file management system such as a disk operating system. The storage is performed in step **205** and typically the image has a unique name on the file storage device or at least a unique full path name for this type of image file. Then according to the present invention, in step **207** a data object is created for each digital image and is stored in a database. This storage is in addition to the storage of the original file for the original image on a file storage device. The storage in the database typically is performed by a picture management system which is typically a separate piece of software which creates and stores the data object for each digital image and also which maintains the database. In one particular embodiment, a lower resolution version of a digital image, such as a "thumbnail" version is stored in the database along with a link to the original image stored on the file storage device under control of the file management system (in step **203**). The link which is stored in association with the thumbnail version refers back to the original image by identifying the picture title or caption as well as the full path name of the original image stored on the file storage device in step **203**. Examples of the thumbnail versions or representations of the original digital image are provided below in this discussion. In step **209**, the user of the computer system then may create

a picture album or another type of media container using a media authoring program of the invention. In one embodiment, the picture album has multiple pages with multiple pictures in the picture album. It will be appreciated that the term digital media includes digital images, such as digital pictures imported or acquired from a digital camera as well as audio files and video files (e.g. Quicktime files) and other digital documents such as word processing files. Thus, a digital picture is a form of a digital media. Similarly, media container is a general term which includes a picture album, but a media container may also include and contain other types of media including audio files or video files or software files such as word processing document files.

In one aspect of the present invention, step **209** of FIG. **4** may be performed substantially automatically under the control of the picture management system which guides a user step by step in a predetermined fashion through a series of questions or steps such that the user is driven from the capture process in step **201** to a publishing process which is described further below. Thus, the user is driven step by step through the capture process of step **201** and the creation of a picture album by selecting a layout and the pictures for the album and then the picture management system transmits the necessary information to another computer system, such as a server computer system, which then automatically generates the viewable pages. These viewable pages are distributable over a network to other client computer systems. Thus, referring back to FIG. **2**, this aspect of the present invention may allow a user on a client computer system **121** to create a media container which contains digital media and publish this media container with its digital media onto the Internet for other client computer systems to be able to view the media container with its digital media. This publication occurs through the use of one computer program on the client computer system which is designed to interact with another computer program on a server computer system, such as server computer system **111**. The interaction between the software on the client computer system, such as system **121** and the server computer system, such as server **111**, is designed such that the media container information and the digital media are supplied from the client computer system to the server computer system which then automatically generates the viewable pages, such as viewable HTML pages. These pages are then made available to a web server **109** which allows other client computer systems, such as client computer system **125** to view the HTML pages which have been published by the client computer system **121**.

The ease of use of this aspect of the present invention is such that with only a few selections on a graphical user interface presented by the picture management software on the client computer system, the user of the client computer system can cause a media container with its associated digital media to be published to the Internet for others to view with conventional web browsers, such as Netscape's Navigator or Microsoft's Internet Explorer. It may only require the user of the client system to make a few selections by pointing and clicking at various radio buttons or icons on the computer screen of the client computer system to cause this publication to occur. For example, this publication may occur by making only one or two selections. It should also be noted that with the present invention any unrestricted member of the public has the capability to publish such media containers with the associated digital media. It will also be appreciated that while the interaction described herein between the client computer system, such as system **121** and the server system **111** is through the web server **109**, it will be appreciated that the client system, such as system **121** may communicate directly with the server system **111** in an alternative embodiment or may communicate with only the web server **109** (where this server also provides the functions of server **111** and is also coupled to a media database, such as database **110**).

FIG. **5** shows an overview of a process according to the present invention for publishing a collection of digital media, typically in a media container, onto a network, such as a network operating according to the HTTP protocol. The method of FIG. **5** begins in step **225** in which images are acquired. These images may be acquired from a digital camera, or a scanner, or from a file storage device such as a CD ROM or a hard disk. Then an album or other media container is authored by selecting a layout and style. Typically in the case of a picture album it will have multiple pages with multiple pictures, and a layout and style will be selected such that the number of pictures per page and the positions of the pictures is determined by the layout and the style. The album is convertible into an Internet viewable format such as the HTML format. In step **227** the software according to the present invention transmits the album format data and signatures (or the actual images themselves in an alternative embodiment) to a server computer system. In a typical embodiment, the album format data specifies the layout and style and number of pictures in the album. The signatures represent the content of each of the images in the media container, in this case a picture album. Further information with respect to the signatures may be obtained from copending U.S. application entitled "Methods and Apparatuses for Transferring Data Between Data Processing Systems" which is filed on the same date herewith by inventors Chan Chiu, Steve Morris, and Wu Wang; this copending application is hereby incorporated herein by reference. The signatures require less time to transmit and allow the client and the server computer systems to avoid transmitting images which already exist at a recipient system. In step **229** of FIG. **5**, the server computer system saves the album format data and images into a database which is described further below. The images are converted into a web-viewable format before any requests to view the images are made by other client computer systems. In this manner, the web-viewable format images are pre-generated prior to "publication" or distribution to other client computer systems. In step **231**, a user on another client computer system uses a web browser to request a view of an album and this request is sent to the server computer system. In one embodiment, the web browser on another client computer system sends a request to the web server **109** which then forwards this request to the server **111** which then determines the requested album and provides this requested album back to the web server **109** in a form which is viewable to the web browser operating on another client computer system, such as client computer system **125**. The server computer system in step **233** thus generates an appropriate page of an album in HTML format and sends the page to the web browser which requested a viewing of the album. The various steps of FIG. **5** will be described further be low in conjunction with FIGS. **6A**, **7**, **8**, and **9**.

FIG. **6A** shows one typical method of performing step **225** of FIG. **5**. It will be appreciated that the sequence. of steps shown in FIG. **6A** is arbitrary and that other sequences are possible depending on the way in which the user interacts with the picture management software on the client system which is publishing an album ("publishing client system"). In step **241**, the user of the publishing client system chooses an album layout and style. This involves picking a particular

layout and style from a set of predefined layouts and styles or in an alternative embodiment allowing a user to provide user-definable layouts and styles. Examples of these styles are shown in FIG. **13**. In step **243**, the user selects images for the various image slots on the album's pages. In step **245**, the user considers whether or not to change the layout or style of the album. The user may at any time change the layout or style and the album will automatically and dynamically reformat itself. If the user decides to change the style or layout, processing returns to step **241**. If no change is desired, then processing proceeds from step **245** to step **247**. At this point, the user may enter a caption or title, if any, for each of the images. In step **249** the user may change the view of an image by zooming, rotating, and/or panning the image. As noted above, the user may perform these various steps in a different sequence. For example, the views of an image may be modified after an image has been selected for a particular slot in step **243**. The captions may be entered after changing the view. It will be appreciated that other sequences for these steps may be performed.

FIG. **6**B illustrates a process for creating a media container, such as a picture album according to the present invention. In step **261**, the picture management system, which may be considered a picture album authoring software, determines an ordered list of pictures for a desired album. Typically, the user will have selected certain pictures for a desired album and these pictures are put in an ordered list. The order of the pictures in the list may be changed by the user. In step **263** the album authoring software determines a selected layout and style for the desired album. This will typically be performed by receiving input from a user, such as input derived from a graphical user interface for the layout and styles such as that shown in FIG. **13**. In step **265**, the album authoring software determines the set of album pages based upon the selected layout. Further, the album authoring software assigns a unique number to each slot on the ordered set of album pages. Then in step **267**, the album authoring software assigns the ordered list of pictures to the numbered slots on the album pages. For example, picture **1** in the ordered list of pictures is placed into slot **1** which would typically be on page **1** of the album. Picture **2** in the ordered list of pictures is placed into slot **2** which may be on page **1** of the album or on page **2** of the album. This assignment is performed for all of the pictures in the ordered list of pictures currently selected by the user. In step **269**, the album authoring software scales each picture if necessary to cause it to fit into the corresponding slot on the album page. The aspect ratio of the picture is maintained after the scaling operation. The scaling operation is performed using conventional scaling techniques which achieve the same aspect ratio after the scaling operation. That is, the aspect ratio (height and width) of the original picture is the same as the aspect ratio of the picture as scaled to fit into the assigned slot on the album page. In step **271**, the album authoring software allows the user to edit the the album. These editing options include adding or deleting pictures, changing the layout, editing each picture (e.g. panning, zooming, and rotating), etc. It will be appreciated that editing of the album may invoke the process of FIG. **6**B to be repeated beginning with step **261** in which the album authoring software determines the ordered list of pictures. For example, if an additional picture is added, the list is changed and an additional album page may be added if an available slot on a page does not exist.

After the user has created a picture album with associated pictures, the user may then decide to publish or otherwise distribute the picture album by making it available for viewing to web browsers over the Internet. The beginning of this process is shown in FIG. **7**, which shows a sequence of steps represented by one step, step **227** of FIG. **5**. In step **281** of FIG. **7**, the client system from which the album will be published logs into a server system via a connection. Typically this connection is via the Internet and thus Internet protocols are used between the client system and the server system. In one embodiment, a client system, such as system **121** communicates to the server computer system **111** through the web server **109** shown in FIG. **2**. The client system from which publication is to occur sends the log-in message, such as user ID and user password to the server system. In step **283**, the server system responds to a log-in request and confirms acceptance. Then in step **285**, the user at the client system selects an album name. This selection may occur by typing in a name or by selecting a name from a list. The client system then transmits the album name to the server system. In step **287**, the server determines whether the selected album name is already used on the server for this user. In one embodiment, referring back to FIG. **2**, the server computer system may be the server system **111** which maintains a picture database for the particular user; this picture database is contained within database **110** and is typically for many users, including the user who logged in in step **281**. If, in step **287**, the server determines that the same album name is already used by this user, then the server sends a message to the client system to ask the client system, in step **289**, whether the client system wants to overwrite the selected album. If the client responds with a no then the client is again requested in step **285** to select a name for an album. If the client system responds to the question from step **289** by indicating yes then processing proceeds from step **289** to step **291**. If the server in step **287** determines that the selected album name is not already used on the server for this user, then processing proceeds directly from step **287** to step **291**. In step **291**, the client system which is publishing the album transmits the album format data to the server and also transmits a signature of each picture in the picture album to the server. The server has its own database of signatures for images of this user and compares the signatures received in step **291** to the signatures stored in a picture database for this user. This comparison allows the server to determine which images are new (e.g. they are not stored on the server for this user). Thus the server determines from this comparison which images are new and provides a list of such images to the client. In step **295**, the client transmits these new images to the server and the server stores the new images and corresponding signatures. The signatures are typically stored in a picture database maintained on the server system as described below.

FIG. **8** shows further processing steps which are performed by the server system after receiving the album format data and the images from the client system. The steps of FIG. **8** correspond to step **229** of FIG. **5**. The server in step **301** determines the number of images, the layout and style, and signatures for each of the images from the album format data. In step **303**, the album format data is saved for this album of this user. Typically, the album format data will be saved in the picture database maintained by the server computer system. In step **305**, the server computer system, such as server system **111** generates images for HTML viewing. In one embodiment, two different JPEG images of different resolutions are generated for HTML viewing. These JPEG images are generated from the original master image which is stored at the server and which was received from the client in step **295** of FIG. **7**. Typically, these two JPEG images will be saved for later HTML viewing. These

images are usually pre-generated prior to any HTML viewing or any distribution or publication over the Internet. The server also saves a reference between these images and the original master image in order to allow downloading of the original master image upon a selection of one of the HTML viewing images.

In one embodiment of the present invention, the picture album is now ready to be viewed and thus be distributed or published over the Internet. A web browser request from another client system, such as client system **125**, is received at the web server **109**, and the web server **109** forwards this request to the server **111**. FIG. **9** illustrates in one example a method for returning the HTML page requested by the other client computer system. The steps shown in FIG. **9** are shown as one step **233** in FIG. **5**. Referring back to FIG. **9**, step **321** involves the server, such as server **111** receiving a URL (universal resource locator) request, from a web browser operating on another client computer system. This request is for viewing of an album. It will be appreciated that the server may receive this request from the client system which published the album. In step **323**, the server obtains the album format data from its database and decomposes this data in order to determine the number of pages, the number of images, the layout information, and other information necessary to generate the album page. The server in step **325** then dynamically generates a formatted HTML page for the requested page (where the first page of the album is the default page) of the album. At this point the HTML page is partially completed (with image URLs specifying pictures in the album pages). In step **327** the server determines the image URL from the images in the partially completed page of the album. The server then obtains a copy of the pre-generated HTML viewable image from storage (e.g. from the server's database). This stored pre-generated HTML viewable image was created in one embodiment in step **305** of FIG. **8**. The HTML viewable images are then placed into their appropriate slots on the HTML album page. Then in step **331**, the server causes the web browser which sent the request in step **321** to display the completed HTML album page to the user of the client system which is operating the web browser. The server then in step **333** transmits an alternative view of an image which was selected. by a user of the web browser. For example, the user may select a particular picture by pointing a cursor on the display at the image and by selecting the image by pressing a button, such as a mouse button, while the cursor is positioned over the image; this selection signals to the server to transmit an alternative view which may be a higher resolution image, such as the second JPEG image created in step **305**.

FIGS. **10** and **11** illustrate examples according to one embodiment of the present invention for two different computer readable storage media. It will be appreciated that the actual memory which stores this information may be different elements, such as the memory **509** and the mass memory **515** or they may be the same element, such as the mass memory **515**. In one example of a network computer where there is no non-volatile mass memory, the necessary software files and data files may be downloaded to the memory **509** for execution in a processor in a network computer. In this case, the memory **509** provides the computer readable storage medium.

FIG. **10** illustrates an example of a computer readable storage medium containing various elements which are used with one embodiment of the present invention. The medium **601** includes a file system and an operating system (OS) element or module **603** which is used to control the file system for the client computer system as well as providing the operating system support such as the disk operating system and other aspects of the operating system. Another element is the album authoring and publishing software **605** which is used to create and modify albums and to interface with the server system in order to publish and/or share those albums. Another element is a dedicated database which is dedicated to the album authoring/publishing software. This database element **607** includes information for the various pictures in the various albums a user may create. While digital pictures represent one embodiment of the present invention, it will be appreciated that digital media or media objects refers generally to audio digital media, video digital media and software files, such as a word processing file created by a word processing computer program. However, the preferred embodiment is one in which the digital media or digital pictures are assembled into a picture album, where the album has multiple pages and where at least some of the pages include multiple pictures.

As shown in FIG. **10**, the file system/OS element **603** includes media objects **621** and **623** which are the actual binary data of two different media objects, media object **1** and media object **2**, stored on a hard disk or other media under control of the disk operating system. The disk operating system creates file system properties, such as properties **622** and **624** which specify various file system related properties for the two media objects. These include file size, date of creation, and document type (e.g. JPEG, BMP, etc.). The album authoring and publishing software element **605** includes four modules **641**, **642**, **643**, and **644**. The signature generator and comparator **641** is the executable computer program for generating and comparing the signatures or representations according to the present invention. The album authoring tools **641** allows a user to create a picture album by selecting layout information which specifies the number and location of pictures on a page throughout multiple pages of a picture album. The interface to server system module **643** includes Internet and web services allowing the client computer system which includes the computer readable memory **601** to interface with a server system, such as the server computer **111** of FIG. **2**. The interface to dedicated database module **644** includes database searching and editing tools allowing the album authoring and publishing software to search and edit the dedicated database **607**.

FIG. **10** shows an example of the dedicated database **607**. It will be appreciated that data may be stored in other formats and ways in this database. As shown in FIG. **10**, for each media object, there is stored in the database a lower resolution version of the digital picture as well as the signature of the media object and a link to the original (higher resolution) media object as well as information indicating the properties of the media object. Thus, the lower resolution and signature of media object **1** is stored with a link to the original media object **621** stored in the file system as well as the properties of the media object **1**, which properties are typically in addition to the file system's properties **622**. Similarly, for media object **2**, there is stored a lower resolution version, such as a thumbnail image, of the media object **2** and the signature or representation of the media object **2**. There is also a link or pointer to the original media object **2** which is the media object **623** maintained by the file system. Further, there are properties for the media object **2** which are in addition to the properties **624**. The database **607** further includes information specifying layout and other information for album **1**, labeled as information **681**, and information **683** specifies information for a second album which may include different pictures than album **1** or may include some of the same pictures as album **1**.

In one embodiment, the client computer system's computer readable media **601** may at some time be entirely stored in non-volatile mass memory, such as a hard disk. At other times, the various elements shown in FIG. **10** may be dispersed between dynamic memory, such as memory **509**, and a mass memory, such as mass memory **515**.

FIG. **11** shows an example of the computer readable storage medium **701** which may be used with a server computer system of the present invention. This memory, which again may be dispersed among memory elements or may be stored entirely on a hard disk or other non-volatile storage media, includes three elements which are the file system/operating system element **703**, the album publishing/sharing software **705**, and the dedicated database for the album publishing software **707**.

The file system and operating system element **703** includes the original, higher resolution media objects **1** and **2** shown as elements **711** and **713**. These elements are the actual digital (or other) data of the media object stored on the computer readable medium under control of the file or storage system such as a disk operating system. The file or storage system also stores properties which are the file system's properties for the media object, such as properties **712** and **714**. These properties typically include the file's size for each media object as well as the date of creation, the date of last modification and the type of document. The album publishing/sharing software **705** includes a signature generator and comparator module which is responsible for generating representations or signatures of the media objects and to compare signatures or representations in accordance with the present invention. The web album publishing interface **719** performs functions relating to decoding information with respect to the albums and generating albums as a result of decoding the information specifying album format. The interface to web server system **721** is an optional software module which is used to allow the server computer system **111** to interface with the web server **109**. Typically, some services are required in order to interface between the album publishing and sharing software and the software required for providing web server functionality. The interface to the dedicated database element **723** provides for database searching and editing of the dedicated database **707**.

The dedicated database **707** includes information **731** for a first album of user **1** and information **733** for a second album of user **1**. It also includes information **735** for a first album of a second user and information **737** for a second album of the second user. There is also stored in the database **707** the signatures for and the links to the original media object for the first user. This information may be stored in separate tables or together in one table. The links point back to an original media object, typically by picture name and full path name to the original media object, such as media object **711** as stored in the file or storage system of the server system. The signatures are used when comparing signatures received from the client system when connected with user **1** in the case of the signatures stored with element **739**. Also for user **1**, the database either stores or refers to a separate storage for the HTML viewing images for all albums of user **1**. This element **743** is generated from the media object, such as the original media object **1** stored as element **711** in the file system. Typically, the HTML viewing images are a lower resolution version of the original media object and will be displayed to users when browsing the web server **109**. The database **707** contains similar information, such as the elements **741** and **745** for the second user.

FIGS. **12**A, **12**B and **12**C illustrate three views of a graphical user interface according to an album authoring and publishing software according to the present invention which may be considered a picture management system. These three figures show a window **1201** having a menu bar **1202** which provides pull-down menus for various commands in the picture management software of one aspect of the present invention. Below the menu bar **1202** is a bar **1204** with icons which represent various commands for the software. Immediately below this bar **1204** is a shoebox window showing a shoebox icon **1203** which is shown as being opened. To the right of this shoebox icon **1203** is a row of thumbnail images **1209**. In one embodiment, these thumbnail images may be created using conventional software, such as that provided by LeadTools which is used to create a bitmap for the thumbnail version of the original image. Below the row of thumbnails **1209** is an album page **1207** in the opened sample album **1205**. A properties window is shown open in all three images (FIGS. **12**A, **12**B, and **12**C). These picture properties windows **1211**, **1211***a*, and **1211***b* show the various properties or attributes saved for a particular picture in a picture database maintained by and for the authoring/publishing software of the present invention. As can be seen from FIGS. **12**A, **12**B, and **12**C, the attributes or properties for each image may include the title of the image, when the picture was taken, where the picture was taken, who was in the picture, the photographer of the picture, text comments, as well as file properties (e.g. file type, number of pixels, file size, path name, date the file was last modified) etc. The picture properties may include, as shown in window **1211***b*, a list of all the albums in which this picture is included. It will be appreciated that the text shown in window **1211** is typically entered by a user; this text may be useful later when searching for a particular picture (e.g. such as searching for all pictures which have Jennifer and Lindsey in the picture).

It will be appreciated that a picture database maintained at a client system may include additional information not required to be maintained in a database for the same picture at a server computer system.

FIG. **13** shows a graphical user interface provided by the album authoring/publishing software of the present invention. Window **1201** is partially covered by the layout and style window **1301**. The layout and style window **1301** allows a user to select various layouts **1303** and various styles **1308**. The preview window **1305** shows the currently selected layout. Thus as shown in FIG. **13**, the preview window **1305** shows a two-picture layout with the picture slots **1306** and **1307** staggered on the album page shown in the preview window **1305**. The user may then select the particular layout by clicking the OK button **1311** or may cancel the selection by clicking the cancel button **1313**. Help may be provided by clicking the help button **1315**. It will be appreciated that the term "click" refers to a well known graphical user interface technique whereby a user positions a cursor, typically with a mouse, over a particular screen object and then depresses a mouse button or some other button on the computer system to select the displayed screen object.

FIGS. **14**A–**14**E also show various graphical user interfaces for an album authoring and publishing software according to the present invention. Window **1401** includes the menu bar **1202** and the bar of icons **1204**. In addition, the shoebox icon **1203** is shown as open which reveals the thumbnails **1209** along a row. As can be seen from FIG. **14**A, not all thumbnails are shown along the row and thus there is a scroll bar along the side to allow the user to display the rest of the thumbnails. The shoebox icon **1203** represents all the pictures in the picture database maintained by the authoring/

publishing software of the present invention. That is, the shoebox represents the picture database at a client system, such as the picture database **607** of FIG. **10**. Below the shoebox icon **1203** is a sample album icon **1205** which is shown as being open. This sample album icon is shown as open because a particular page from this sample album is displayed. In particular, page **1207** from the sample album represented by icon **1205** is shown below the row of thumbnails **1209**. The page **1207** includes two digital pictures **1415** and **1417** displayed in a two-picture layout. Page **1** is indicated by page indicator **1208** as shown. Page **2** may be selected by clicking on the tab **1407** or by clicking on either arrow **1409** or **1411**. The user may toggle between these two pages by alternatively clicking on tabs **1405** and **1407**.

FIG. **14**B shows an alternative view of a graphical user interface for an album authoring/publishing software according to one aspect of the present invention. The window **1421** includes an alternative view of the thumbnails **1423** which is to the right of the shoebox icon **1203** and the sample album icon **1425**. The shoebox icon is shown as being open, thus revealing all the thumbnails in the picture database. The sample album icon **1425** is shown as closed; if it were open, as will be clear from the discussion below, only the thumbnail s from this album would be presented in the thumbnail region to the right of the sample album icon **1425**. An album page from the sample album is shown in the lower half of the window **1421**. The sample album icon **1205** is shown as open and a second page of this album is shown to the right (in the album page region). The page indicator **1208** shows that the second page is being displayed for the sample album. Two picture slots **1433** and **1435** remain to be filled on the second page of this album. The user may return to the first page by either clicking on the tab **1405** or clicking on the arrows **1429** or **1427**. It will be appreciated that arrow **1429** moves to the prior page, and arrow **1409** in FIG. **14**A moves to the next page. Arrow **1427** moves to the first page of the album, and arrow **1411** moves to the last page of the album. In FIG. **14**B, page **1431** is the last page of the album as shown by the indicator **1208**.

FIG. **14**C shows an alternative view of a graphical user interface for an album authoring/publishing software according to one aspect of the present invention. The view shown in window **1451** is of thumbnails only. In the particular view shown in FIG. **14**C, all the thumbnails **1453** in the database represented by the shoebox icon **1203** are shown in the window **1451**. The sample album icon **1425** is shown as closed because thumbnails for only that sample album are not being shown.

FIG. **14**D shows yet another graphical interface for an album authoring/publishing software according to one aspect of the present invention. In the window **1461**, and album page only view is shown. That is, no thumbnails are shown in this view. Thus a page from the opened sample album represented by sample album icon **1205** is shown. This page **1207** includes two digital pictures **1415** and **1407** arranged in a two-picture layout.

FIG. **14**E shows yet another graphical user interface for an album authoring/publishing software program according to one aspect of the present invention. The window **1471** shows the concurrent view of thumbnails and a page album. Thus thumbnails **1473** are shown in the upper portion of this window and an album page **1475** is shown in the lower portion of this window. The sample album icon **1205** is opened indicating that a sample page from this album is shown, in this case page **1475**. This particular sample album has three pages as indicated by the indicator **1208** and by the tab labeled with a "3" as shown in FIG. **14**E. The shoebox icon **1203***a* is shown as closed and the sample album icon **1425***a* is shown as open. This indicates that the thumbnails **1473** are only from the sample album selected by selecting the icon **1425***a*. In this manner, the thumbnails displayed on the upper portion of window **1471** (the thumbnail display region) may be from an album which is different than the album shown in the lower portion of the window **1471**. In this manner, a user may copy an image from one album into another album.

The foregoing description has provided numerous examples of the present invention. It will be appreciated that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims. One modification involves a client software program which can automatically generate publishable web pages (HTTP format) without any server software (at the server) which is designed to work specifically with the client software. This client software would allow a client user to turn virtually any ISP's web server into a web publishing server without any specialized server software at the server. This modification is similar to the distributed client and server software described above except that the client software automatically generates the HTML pages and links to the images and then performs an intelligent FTP file transfer to the ISP's web server. The intelligent FTP file transfer places the transferred files into the proper directories at the ISP's web server. Typically, this may require the assistance or permission of the ISP. Further details concerning particular embodiments of the present invention may also be found in the following copending patent applications which were filed on the same date as this application and which are hereby incorporated herein by reference; these copending applications are as follows: "Methods and Apparatuses for Transferring Data Between Data Processing Systems" by inventors Chan Chiu, Steve Morris, and Wu Wang; and "Methods and Apparatuses for Acquiring a Digital Image for Use in a Digital Processing System" by inventors James Lei and Wu Wang.

What is claimed is:

1. A method for publishing on a network operating according to an HTTP protocol a collection of digital media, said method comprising:

generating a collection of digital media at a client digital processing system;

transmitting collection information, which describes said collection of digital media, from said client digital processing system to a server digital processing system;

pregenerating automatically from said collection information a plurality of presentable media each of which is capable of being presented to other client digital processing systems which are coupled to said network which is operating according to said HTTP (hypertext transfer protocol) protocol.

2. A method as in claim **1** wherein said digital media comprises digital images and wherein said presentable media comprises viewable images and wherein said client digital processing system and said server digital processing system are each programmed to interact together such that said server digital processing system generates automatically, from said collection information, said plurality of viewable images.

3. A method as in claim **2** further comprising:

transmitting, from said server digital processing system, said plurality of viewable images to a World Wide Web (WWW) server digital processing system;

transmitting said plurality of viewable images from said WWW server digital processing system to at least one of said other client digital processing systems, wherein said WWW server digital processing system is coupled to said network and wherein said client digital processing system is coupled to said network and transmits said collection information to said server digital processing system using said HTTP protocol.

4. A method as in claim **2** wherein each of said plurality of viewable images is in an HTML (hypertext markup language) and wherein a user of said client digital processing system does not need to create or convert said collection of digital images into said HTML format.

5. A method as in claim **4** wherein said collection is an album of digital images, said album having a layout and a style specified by a user of said client digital processing system.

6. A method as in claim **5** wherein said network is one of an Internet or an intranet.

7. A method as in claim **5** wherein said plurality of viewable images is not persistently maintained at said server digital processing system.

8. A method as in claim **7** wherein each of said plurality of images includes at least one digital picture, and wherein each of said at least one digital picture is persistently maintained at said server digital processing system.

9. A method as in claim **8** wherein said server digital processing system generates one of said viewable images and retrieves from storage at least one of said digital picture when it receives a web browsing request from one of said other client digital processing systems, and wherein after said other client digital processing system stops viewing said one of said viewable images, said one of said viewable images is not retained at said server digital processing system.

10. A method as in claim **8** wherein each of said at least one digital picture is generated and stored at said server digital processing system prior to any browsing request, from said other client digital processing systems, to view any one of said each of said at least one digital picture.

11. A method as in claim **2** wherein a user of said client digital processing, after having created said collection of digital images, causes said collection to be published by a simple selection.

12. A method as in claim **11** wherein said simple selection comprises one of a single selection using a pointing device coupled to said client digital processing system or two selections using said pointing device or three selections using said pointing device.

13. A method as in claim **2** wherein a user of said client digital processing systems may be an unrestricted public member.

14. A method of claim **5** wherein each of said plurality of viewable images is capable of being viewed by using a web browser at one of said other client digital processing systems to request an HTML page which displays a page of said album.

15. A method of claim **2** wherein said client digital processing system stores a copy of each of said digital images on a client file storage device which is managed by a client file management system and said client digital processing system stores in a client database a first representation and a second representation of said copy of each of said digital images and stores in said database an association to said copy of each of said digital images.

16. A method as in claim **15** wherein said first representation comprises a lower resolution version of said copy and said second representation comprises a digest of contents of said copy and said association comprises a reference to said copy.

17. A method as in claim **2** wherein said server digital processing system stores a copy of each of said digital images on a server file storage device which is managed by a server file management system and said server digital processing system stores in a server database a first representation and a second representation of said copy of each of said digital images and stores in said database an association to said copy of each of said digital images.

18. A method as in claim **17** wherein said first representation is an HTML viewable version of said copy and said second representation comprises a digest of contents of said copy and said association comprises a reference to said copy in said server file storage device.

19. A distributed computer readable storage medium containing executable computer program instructions which when executed cause a client digital processing system and a server digital processing system to perform a method comprising:

generating a collection of digital media at a client digital processing system;

transmitting collection information, which describes said collection of digital media, from said client digital processing system to said server digital processing system;

pregenerating automatically from said collection information a plurality of presentable media each of which is capable of being presented to other client digital processing systems which are coupled to a network which is operating according to an HTTP (hypertext transfer protocol) protocol.

20. A distributed computer readable storage medium as in claim **19** wherein said digital media comprises digital images and wherein said presentable media comprises viewable images and wherein said client digital processing system and said server digital processing system are each programmed to interact together such that said server digital processing system generates automatically, from said collection information, said plurality of viewable images.

21. A distributed computer readable storage medium as in claim **20**, wherein said method further comprises:

transmitting, from said server digital processing system, said plurality of viewable images to a World Wide Web (WWW) server digital processing system;

transmitting said plurality of viewable images from said WWW server digital processing system to at least one of said other client digital processing systems, wherein said WWW server digital processing system is coupled to said network and wherein said client digital processing system is coupled to said network and transmits said collection information to said server digital processing system using said HTTP protocol.

22. A distributed computer readable storage medium as in claim **20** wherein each of said plurality of viewable images is in an HTML (hypertext markup language) and wherein a user of said client digital processing system does not need to create or convert said collection of digital images into said HTML format.

23. A distributed computer readable storage medium as in claim **22** wherein said collection is an album of digital images, said album having a layout and a style specified by a user of said client digital processing system.

24. A distributed computer readable storage medium as in claim **23** wherein said network is one of an Internet or an intranet.

25. A distributed computer readable storage medium as in claim 23 wherein said plurality of viewable images is not persistently maintained at said server digital processing system.

26. A distributed computer readable storage medium as in claim 25 wherein each of said plurality of images includes at least one digital picture, and wherein each of said at least one digital picture is persistently maintained at said server digital processing system.

27. A distributed computer readable storage medium as in claim 26 wherein said server digital processing system generates one of said viewable images and retrieves from storage at least one of said digital picture when it receives a web browsing request from one of said other client digital processing systems, and wherein after said other client digital processing system stops viewing said one of said viewable images, said one of said viewable images is not retained at said server digital processing system.

28. A distributed computer readable storage medium as in claim 26 wherein each of said at least one digital picture is generated and stored at said server digital processing system prior to any browsing request, from said other client digital processing systems, to view any one of said each of said at least one digital picture.

29. A distributed computer readable storage medium as in claim 20 wherein a user of said client digital processing, after having created said collection of digital images, causes said collection to be published by a simple selection.

30. A distributed computer readable storage medium as in claim 29 wherein said simple selection comprises one of a single selection using a pointing device coupled to said client digital processing system or two selections using said pointing device or three selections using said pointing device.

31. A distributed computer readable storage medium as in claim 20 wherein a user of said client digital processing systems may be an unrestricted public member.

32. A distributed computer readable storage medium of claim 23 wherein each of said plurality of viewable images is capable of being viewed by using a web browser at one of said other client digital processing systems to request an HTML page which displays a page of said album.

33. A distributed computer readable storage medium of claim 20 wherein said client digital processing system stores a copy of each of said digital images on a client file storage device which is managed by a client file management system and said client digital processing system stores in a client database a first representation and a second representation of said copy of each of said digital images and stores in said database an association to said copy of each of said digital images.

34. A distributed computer readable storage medium as in claim 33 wherein said first representation comprises a lower resolution version of said copy and said second representation comprises a digest of contents of said copy and said association comprises a reference to said copy.

35. A distributed computer readable storage medium as in claim 20 wherein said server digital processing system stores a copy of each of said digital images on a server file storage device which is managed by a server file management system and said server digital processing system stores in a server database a first representation and a second representation of said copy of each of said digital images and stores in said database an association to said copy of each of said digital images.

36. A server digital processing system for publishing, on a network operating according to an HTTP protocol, a collection of digital media, said server digital processing system comprising:

a processor;

a network interface coupled to said network and coupled to said processor, said network interface receiving collection information which describes a collection of digital media from a remotely located client digital processing system;

a file storage device coupled to said processor, said file storage device storing copies of said digital media under control of a file management system, and wherein said processor pregenerates automatically from said collection information a plurality of presentable media each of which is capable of being presented by other remotely located client digital processing systems which are coupled to said network.

37. A server digital processing system as in claim 36 wherein said digital media comprises digital images and wherein said presentable media comprises viewable images and wherein said processor maintains a database and creates web-viewable versions of each of said plurality of viewable images, and wherein said database comprises for each of said web-viewable versions an association between said each of said web-viewable versions and said corresponding copy of said digital image.

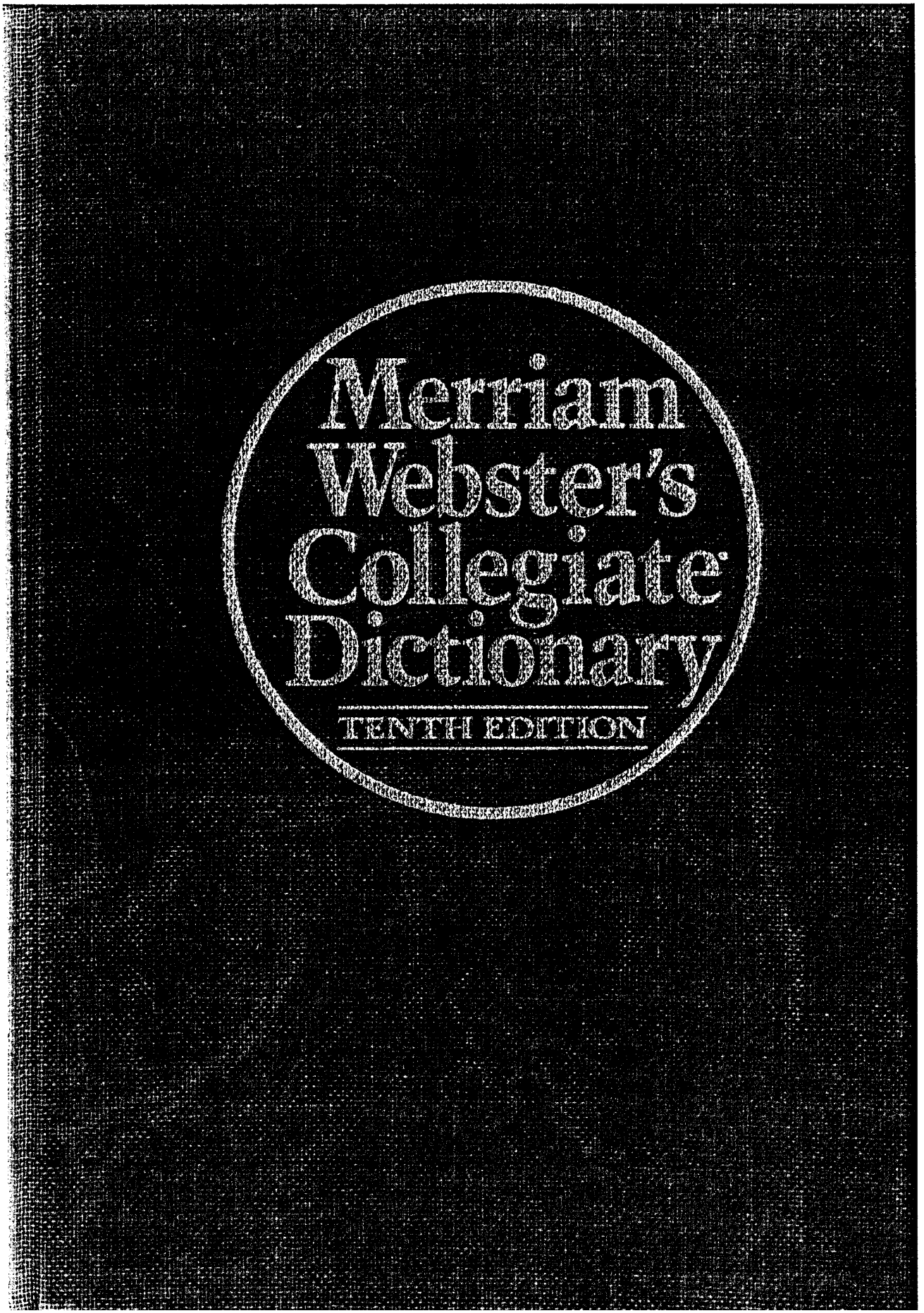
38. A computer readable storage medium containing executable computer program instructions which when executed cause a client digital processing system and a server digital processing system to perform a method comprising:

generating a collection of digital media at a client digital processing system;

transmitting collection information, which describes said collection of digital media, from said client digital processing system to said server digital processing system;

pregenerating automatically from said collection information a plurality of presentable media each of which is capable of being presented to other client digital processing systems which are coupled to a network which is operating according to an HTTP (hypertext transfer protocol) protocol.

* * * * *


Merriam
Webster's
Collegiate
Dictionary
TENTH EDITION

Merriam-Webster's Collegiate® Dictionary

A
B

TENTH EDITION

Merriam-Webster, Incorporated
Springfield, Massachusetts, U.S.A.

A GENUINE MERRIAM-WEBSTER

The name *Webster* alone is no guarantee of excellence. It is used by a number of publishers and may serve mainly to mislead an unwary buyer.

Merriam-Webster™ is the name you should look for when you consider the purchase of dictionaries or other fine reference books. It carries the reputation of a company that has been publishing since 1831 and is your assurance of quality and authority.

Library of Congress Cataloging in Publication Data
Main entry under title:

Merriam-Webster's collegiate dictionary. — 10th ed.
p. cm.
Includes index.
ISBN 0-87779-708-0 (unindexed : alk. paper). — ISBN 0-87779-709-9 (indexed : alk. paper). — ISBN 0-87779-710-2 (deluxe : alk. paper). — ISBN 0-87779-707-2 (laminated cover).
1. English language—Dictionaries. I. Merriam-Webster, Inc.
PE1628.M36 1997
423—dc20 96-42529
CIP

Merriam-Webster's Collegiate® Dictionary, Tenth Edition principal copyright 1993

COLLEGIATE is a registered trademark of Merriam-Webster, Incorporated

Made in the United States of America

1920RMcN97

(1542) : contrary to nature, reason, or common sense : ABSURD — **pre·pos·ter·ous·ly** *adv* — **pre·pos·ter·ous·ness** *n*

pre·po·ten·cy \(,)prē-'pō-tᵊn(t)-sē\ *n* (1646) **1** : the quality or state of being prepotent : PREDOMINANCE **2** : unusual ability of an individual or strain to transmit its characters to offspring because of homozygosity for numerous dominant genes

pre·po·tent \-tᵊnt\ *adj* [ME, fr. L *praepotent-, praepotens*, fr. *prae-* + *potens* powerful — more at POTENT] (15c) **1 a** : having exceptional power, authority, or influence **b** : exceeding others in power **2** : exhibiting genetic prepotency — **pre·po·tent·ly** *adv*

¹prep·py *or* **prep·pie** \'pre-pē\ *n, pl* **prep·pies** [*prep*] (1967) **1** : a student at or a graduate of a preparatory school **2** : a person deemed to dress or behave like a preppy

²preppy *or* **preppie** *adj* (1967) **1** : relating to, characteristic of, or being a preppy **2** : relating to or being a style of dress characterized esp. by classic clothing and neat appearance — **prep·pi·ly** \'pre-pə-lē\ *adv* — **prep·pi·ness** \'pre-pē-nəs\ *n*

pre·pran·di·al \(,)prē-'pran-dē-əl\ *adj* (1822) : of, relating to, or suitable for the time just before dinner ⟨a ~ drink⟩

pre·preg \,prē-'preg, 'prē-,\ *n* [*pre-* + *impregnated*] (1954) : a reinforcing or molding material (as paper or glass cloth) already impregnated with a synthetic resin

¹pre·print \'prē-,print, ,prē-'print\ *n* (1889) **1** : an issue of a technical paper often in preliminary form before its publication in a journal **2** : something (as an advertisement) printed before the rest of the publication in which it is to appear

²pre·print \(,)prē-'print\ *vt* (1926) : to print in advance for later use

pre·pro·cess \(,)prē-'prä-,ses, -'prō-, -səs\ *vt* (1942) : to do preliminary processing of (as data) — **pre·pro·ces·sor** \-,se-sər, -sə-sər, -,sȯr\ *n*

pre·pro·fes·sion·al \,prē-prə-'fesh-nəl, -'fe-shə-nᵊl\ *adj* (1926) : of or relating to the period preceding specific study for or practice of a profession

prep school *n* (1895) : PREPARATORY SCHOOL

pre·pu·ber·al \(,)prē-'pyü-b(ə-)rəl\ *adj* (ca. 1935) : PREPUBERTAL

pre·pu·ber·tal \-bər-tᵊl\ *adj* (1859) : of or relating to prepuberty

pre·pu·ber·ty \-bər-tē\ *n* (1922) : the period immediately preceding puberty

pre·pu·bes·cence \,prē-pyü-'be-sᵊn(t)s\ *n* (1916) : PREPUBERTY

pre·pu·bes·cent \-sᵊnt\ *adj* (1904) : PREPUBERTAL — **prepubescent** *n*

pre·puce \'prē-,pyüs\ *n* [ME, fr. MF, fr. L *praeputium*] (15c) : FORESKIN; *also* : a similar fold investing the clitoris — **pre·pu·tial** \prē-'pyü-shəl\ *adj*

pre·quel \'prē-kwəl\ *n* [*pre-* + *-quel* (as in *sequel*)] (1972) : a literary or dramatic work whose story precedes that of an earlier work

Pre–Ra·pha·el·ite \(,)prē-'ra-fē-ə-,līt, -'rā-, -'rä-\ *n* (1850) **1 a** : a member of a brotherhood of artists formed in England in 1848 to restore the artistic principles and practices regarded as characteristic of Italian art before Raphael **b** : an artist or writer influenced by this brotherhood **2** : a modern artist dedicated to restoring early Renaissance ideals or methods — **Pre–Raphaelite** *adj* — **Pre–Ra·pha·el·it·ism** \-,li-,ti-zəm\ *n*

pre·reg·is·tra·tion \,prē-,re-jə-'strā-shən\ *n* (1967) : a special registration (as for returning students) prior to an official registration period — **pre·reg·is·ter** \(,)prē-'re-jə-stər\ *vi*

pre·req·ui·site \(,)prē-'re-kwə-zət\ *n* (1633) : something that is necessary to an end or to the carrying out of a function — **prerequisite** *adj*

pre·rog·a·tive \pri-'rä-gə-tiv\ *n* [ME, fr. MF & L: MF, fr. L *praerogativa*, Roman century voting first in the comitia, privilege, fr. fem. of *praerogativus* voting first, fr. *praerogatus*, pp. of *praerogare* to ask for an opinion before another, fr. *prae-* + *rogare* to ask — more at RIGHT] (15c) **1 a** : an exclusive or special right, power, or privilege: as (1) : one belonging to an office or an official body (2) : one belonging to a person, group, or class of individuals (3) : one possessed by a nation as an attribute of sovereignty **b** : the discretionary power inhering in the British Crown **2** : a distinctive excellence — **pre·rog·a·tived** \-tivd\ *adj*

¹pres·age \'pre-sij, *also* pri-'sāj\ *n* [ME, fr. L *praesagium*, fr. *praesagus* having a foreboding, fr. *prae-* + *sagus* prophetic — more at SEEK] (14c) **1** : something that foreshadows or portends a future event : OMEN **2** : an intuition or feeling of what is going to happen in the future **3** *archaic* : PROGNOSTICATION **4** : warning or indication of the future — **pre·sage·ful** \pri-'sāj-fəl\ *adj*

²pre·sage \'pre-sij, pri-'sāj\ *vb* **pre·saged; pre·sag·ing** *vt* (1562) **1** : to give an omen or warning of : FORESHADOW **2** : FORETELL, PREDICT ~ *vi* : to make or utter a prediction — **pre·sag·er** *n, obs*

pre·sanc·ti·fied \(,)pre-'saŋ(k)-ti-,fid\ *adj* (1758) : consecrated at a previous service — used of eucharistic elements

pres·by·ope \'prez-bē-,ōp; 'pres-bē-, -pē-\ *n* [prob. fr. F, fr. Gk *presbys* old man + *ōps* eye — more at EYE] (ca. 1857) : one affected with presbyopia

pres·by·o·pia \,prez-bē-'ō-pē-ə, ,pres-\ *n* [NL] (1793) : a visual condition which becomes apparent esp. in middle age and in which loss of elasticity of the lens of the eye causes defective accommodation and inability to focus sharply for near vision — **pres·by·o·pic** \-'ō-pik, -'ä-\ *adj or n*

pres·by·ter \'prez-bə-tər, 'pres-\ *n* [LL, elder, priest, fr. Gk *presbyteros*, compar. of *presbys* old man, elder; akin to Gk *pro* before and Gk *bainein* to go — more at FOR, COME] (1597) **1** : a member of the governing body of an early Christian church **2** : a member of the order of priests in churches having episcopal hierarchies that include bishops, priests, and deacons **3** : ELDER 4b — **pres·byt·er·ate** \prez-'bi-tə-rət, pres-, -,rāt\ *n*

¹pres·by·te·ri·al \,prez-bə-'tir-ē-əl, ,pres-\ *adj* (ca. 1600) : of or relating to presbyters or a presbytery — **pres·by·te·ri·al·ly** \-ē-ə-lē\ *adv*

²presbyterial *n, often cap* (1928) : an organization of Presbyterian women associated with a presbytery

¹Pres·by·te·ri·an \-ē-ən\ *n* (1640) : a member of a Presbyterian church

²Presbyterian *adj* (1641) **1** *often not cap* : characterized by a graded system of representative ecclesiastical bodies (as presbyteries) exercising legislative and judicial powers **2** : of, relating to, or constituting a Protestant Christian church that is presbyterian in government and traditionally Calvinistic in doctrine — **Pres·by·te·ri·an·ism** \-ē-ə-,ni-zəm\ *n*

pres·by·tery \'prez-bə-,ter-ē, 'pres-, -bə-trē\ *n, pl* **-ter·ies** [ME & LL; ME *presbytory* part of church reserved for clergy, fr. LL *presbyterium* group of presbyters, part of church reserved for clergy, fr. Gk *presbyterion* group of presbyters, fr. *presbyteros* elder, priest] (15c) **1** : the part of a church reserved for the officiating clergy **2** : a ruling body in presbyterian churches consisting of the ministers and representative elders from congregations within a district **3** : the jurisdiction of a presbytery **4** : the house of a Roman Catholic parish priest

¹pre·school \'prē-,skül, (,)prē-'\ *adj* (1914) : of, relating to, or constituting the period in a child's life from infancy to the age of five or six that ordinarily precedes attendance at elementary school

²pre·school \'prē-,skül\ *n* (ca. 1925) : NURSERY SCHOOL, KINDERGARTEN

pre·school·er \-,skü-lər\ *n* (1946) **1** : a child not yet old enough for school **2** : a child attending a preschool

pre·science \'pre-sh(ē-)ən(t)s, 'prē-, -s(ē-)ən(t)s\ *n* [ME, fr. LL *praescientia*, fr. L *praescient-, praesciens*, prp. of *praescire* to know beforehand, fr. *prae-* + *scire* to know — more at SCIENCE] (14c) : foreknowledge of events: **a** : divine omniscience **b** : human anticipation of the course of events : FORESIGHT — **pre·scient** \-sh(ē-)ənt, -s(ē-)ənt\ *adj* — **pre·scient·ly** *adv*

pre·sci·en·tif·ic \,prē-,sī-ən-'ti-fik\ *adj* (1858) : of, relating to, or having the characteristics of a period before the rise of modern science or a state prior to the application of the scientific method

pre·scind \pri-'sind\ *vb* [L *praescindere* to cut off in front, fr. *prae-* + *scindere* to cut — more at SHED] *vi* (1650) : to withdraw one's attention ~ *vt* : to detach for purposes of thought

pre·score \(,)prē-'skōr, -'skȯr\ *vt* (1937) : to record (as sound) in advance for use when the corresponding scenes are photographed in making movies

pre·scribe \pri-'skrīb\ *vb* **pre·scribed; pre·scrib·ing** [ME, fr. L *praescribere* to write at the beginning, dictate, order, fr. *prae-* + *scribere* to write — more at SCRIBE] *vi* (15c) **1** : to lay down a rule : DICTATE **2** [ME, fr. ML *praescribere*, fr. L, to write at the beginning] : to claim a title to something by right of prescription **3** : to write or give medical prescriptions **4** : to become by prescription invalid or unenforceable ~ *vt* **1 a** : to lay down as a guide, direction, or rule of action : ORDAIN **b** : to specify with authority **2** : to designate or order the use of as a remedy — **pre·scrib·er** *n*

pre·script \'prē-,skript, pri-'\ *adj* [ME, fr. L *praescriptus*, pp.] (ca. 1540) : prescribed as a rule — **pre·script** \'prē-,skript\ *n*

pre·scrip·tion \pri-'skrip-shən\ *n* [partly fr. ME *prescripcion* establishment of a claim, fr. MF *prescription*, fr. LL *praescription-, praescriptio*, fr. L, act of writing at the beginning, order, limitation of subject matter, fr. *praescribere;* partly fr. L *praescription-, praescriptio* order] (14c) **1 a** : the establishment of a claim of title to something under common law usu. by use and enjoyment for a period fixed by statute **b** : the right or title acquired under common law by such possession **2** : the process of making claim to something by long use and enjoyment **3** : the action of laying down authoritative rules or directions **4 a** : a written direction for a therapeutic or corrective agent; *specif* : one for the preparation and use of a medicine **b** : a prescribed medicine **c** : something like a doctor's prescription ⟨~s for economic recovery⟩ **5 a** : ancient or long continued custom **b** : a claim founded upon ancient custom or long continued use **6** : something prescribed as a rule

prescription drug *n* (1951) : a drug that can be obtained only by means of a physician's prescription

pre·scrip·tive \pri-'skrip-tiv\ *adj* (1748) **1** : serving to prescribe ⟨~ rules of usage⟩ **2** : acquired by, founded on, or determined by prescription or by long-standing custom — **pre·scrip·tive·ly** *adv*

pre·se·lect \,prē-sə-'lekt\ *vt* (ca. 1859) : to choose in advance usu. on the basis of a particular criterion — **pre·se·lec·tion** \-'lek-shən\ *n*

pre·sell \(,)prē-'sel\ *vt* **-sold** \-'sōld\; **-sell·ing** (1947) **1** : to precondition (as a customer) for subsequent purchase or create advance demand for (as a product) esp. through marketing strategies **2** : to sell in advance ⟨raised money to publish the book by *preselling* film rights⟩

pres·ence \'pre-zᵊn(t)s\ *n* (14c) **1** : the fact or condition of being present **2 a** : the part of space within one's immediate vicinity **b** : the neighborhood of one of superior esp. royal rank **3** *archaic* : COMPANY 2a **4** : one that is present: as **a** : the actual person or thing that is present **b** : something present of a visible or concrete nature **5 a** : the bearing, carriage, or air of a person; *esp* : stately or distinguished bearing **b** : a quality of poise and effectiveness that enables a performer to achieve a close relationship with his audience **6** : something (as a spirit) felt or believed to be present

presence of mind (1665) : self-control so maintained in an emergency or in an embarrassing situation that one can say or do the right thing

¹pres·ent \'pre-zᵊnt\ *n* [ME, fr. OF, fr. *presenter*] (13c) : something presented : GIFT

²pre·sent \pri-'zent\ *vb* [ME, fr. OF *presenter*, fr. L *praesentare*, fr. *praesent-, praesens*, adj.] *vt* (14c) **1 a** (1) : to bring or introduce into the presence of someone esp. of superior rank or status (2) : to introduce socially **b** : to bring (as a play) before the public **2** : to make a gift to **3** : to give or bestow formally **4 a** : to lay (as a charge) before a court as an object of inquiry **b** : to bring a formal public charge, indictment, or presentment against **5** : to nominate to a benefice **6 a** : to offer to view : SHOW **b** : to bring to one's attention ⟨this ~s a problem⟩ **7** : to act the part of : PERFORM **8** : to aim, point, or direct (as a weapon) so as to face something or in a particular direction ~ *vi* **1** : to present a weapon **2** : to become manifest **3** : to come forward as a patient **4** : to make a presentation *syn* see GIVE — **pre·sent·er** *n*

³pres·ent \'pre-zᵊnt\ *adj* [ME, fr. OF, fr. L *praesent-, praesens*, fr. prp. of *praeesse* to be before one, fr. *prae-* pre- + *esse* to be — more at IS] (14c) **1** : now existing or in progress **2 a** : being in view or at hand **b** : existing in something mentioned or under consideration **3** : constituting the one actually involved, at hand, or being considered **4** : of, relating to, or constituting a verb tense that is expressive of pres-